2004

Sixth Workshop on Education in Computer Security: Avoiding Fear, Uncertainty and Doubt Through Effective Security Education

Sixth Workshop on Education in Computer Security

WECS 6

AVOIDING FEAR, UNCERTAINTY AND DOUBT
Through Effective Security Education

NAVAL POSTGRADUATE SCHOOL

The Center for Information Systems Security Studies and Research

AVOIDING FEAR, UNCERTAINTY AND DOUBT

AVOIDING FEAR, UNCERTAINTY AND DOUBT
Through Effective Security Education

Workshop Chair

Cynthia Irvine

Edited by

Cynthia Irvine and Matthew Rose
Naval Postgraduate School, Monterey, California, USA

Center for Information Systems Security Studies and Research
Naval Postgraduate School
Monterey, California

Sixth Workshop on Education in Computer Security (WECS6)

Naval Postgraduate School, Monterey, California
Tutorial Section: 12-14, July 2004
Workshop Section: 15-16 July 2004

WECS is dedicated to the advancement of Information Assurance education. The tutorials are for the advancement of IA education at the undergraduate level with emphasis on reaching under-represented groups. CISR launched WECS in 1999 and is proud to have started such a successful tradition of outreach to the computer security community.

The goal of WECS6 is to bring together members of the information assurance education community to promote an exchange of ideas related to shared educational objectives.

Workshop Chair:
Cynthia Irvine
Computer Science Department
Naval Postgraduate School

Organizing Committee:
Cynthia Irvine (Naval Postgraduate School)
Naomi Falby (Naval Postgraduate School)
J.D. Fulp (Naval Postgraduate School)
Matthew Rose (Naval Postgraduate School)
Daniel Warren (Naval Postgraduate School)

Program Committee: (in alphabetic order)
Blaine Burnham (University of Nebraska at Omaha)
George Dinolt (Naval Postgraduate School)
Deborah Frincke (Univ. of Idaho and Pacific Northwest National Laboratory)
Cynthia Irvine (Naval Postgraduate School)
Tim Levin (Naval Postgraduate School)
Bill Murray (Naval Postgraduate School)
Giovanni Vigna (UC Santa Barbara)

Library of Congress Cataloging-in-Publication Data

A C.I.P. Catalogue record for this book is available from the Library of Congress.

Avoiding Fear, Uncertainty and Doubt
Through Effective Security Education

Edited by Cynthia Irvine and Matthew Rose
ISBN 0-9755139-0-7

Printed in the United States of America at a Kinko's.

Please visit: http://cisr.nps.navy.mil

Contents

Sixth Workshop on Education in Computer Security (WECS6) v

Contributing Authors ix

Preface 1

Security Education for Business 3
WILLIAM HUGH MURRAY, CISSP

teaching computer security to undergraduates 5
RAHUL V. TIKEKAR

A Note Regarding Covert Channels 11
TIMOTHY E. LEVIN AND PAUL C. CLARK

Capture-the-Flag: Learning Computer Security Under Fire 17
LCDR CHRIS EAGLE, AND JOHN L. CLARK

Security as a Component of CS Curriculum at at Liberal Arts College 23
EVERETT L. BULL, JR.

Teaching Context in Information Security 29
MATT BISHOP

Topics in Computer Security 37
JIM GRIFFIN 37

Expressing an IS Policy Within a Security Simulation Game 43
CYNTHIA E. IRVINE AND MICHAEL F. THOMPSON

Honeynet Solutions 51
RONALD C DODGE JR, RICHARD T BROWN, DANIEL J RAGSDALE

The Advanced Course in Engineering on Cyber Security 57
KAMAL JABBOUR AND SUSAN OLDER

The Bastion Network Project 65
J. D. FULP

Practical Security 73
CORRINNE SANDE

Report: WECS5 follow-up 79
PARKER SWANSON

Follow-up report for WECS5 83
JILL M. SNYDER

Report on WECS5 87
DIANE PANNELL

Contributing Authors

William Hugh Murray, Rahul V. Tikekar, Timothy E. Levin, Paul C. Clark, LCDR Chris Eagle, John L. Clark, Everett L. Bull, Jr., Matt Bishop, Jim Griffin, Cynthia E. Irvine, Michael F. Thompson, Ronald C Dodge JR, Richard T Brown, Daniel J Ragsdale, Kamal Jabbour, Susan Older, J. D. Fulp, Corrinne Sande, Parker Swanson, Jill M. Snyder, and Diane Pannell

Preface

Dear Friends and Colleagues in Information Security Education,

What sells television shows, magazines, and newspapers? If the wait on the grocery store checkout line is any indicator, articles telling us that everything is fine do not. Instead, the mass media are full of stories preying upon the public's feelings of fear, uncertainty, and doubt. Tales and reports of the dangers and quandaries encountered by others stimulate the imagination and in their lowest form feed a kind of prurient desire for excitement. We read about sweet innocent victims of kidnap and murder, the downfall of princes of state and industry, of medications once thought reliable and now found harmful, and of the dangers of too much of this and too little of that. The list is endless.

Now added to our daily diet of excitement are cyber security scares. Hardly a day passes without some discussion of a new attack or vulnerability to computer systems. Sometimes these attacks are extensive and highly disruptive and at other times we are notified of system vulnerabilities that will lead to attacks unless "someone does something!" Thus we have entered a climate that is ripe for over-reaction should some massive attack occur. In the aftermath of an attack, the public might choose to reject technology or might accept mechanical controls that might undermine fundamental tenets of our civil society.

Knowledge can free us from superstition and provide a basis for creating solutions to a wide range of cyber security problems. It can help us to ensure that our behavior and laws account for cyber space, but are not the result of some knee-jerk reaction to a cyber attack.

Naturally, information security educators play an important role in addressing the problems of fear, uncertainty and doubt. We can provide our students with an understanding of system vulnerabilities, the threat agents to whom such vulnerabilities would be attractive, and techniques for mitigating those threats. Even more importantly, our students can be practiced in the critical thinking skills necessary to discern cyber security snake oil and voodoo from sound security architectures and products.

It is within this context that we welcome you to the Sixth Workshop on Education in Computer Security (WECS). Our theme this year is "Avoiding Fear, Uncertainty, and Doubt through Effective Security Education." The papers contained in this volume present tools and techniques that have been used in undergraduate and graduate settings. Some of the papers describe entire programs or courses, while others present laboratory activities; a few papers explore rather unusual techniques for conveying the cyber security message. The scope of our field continues to expand and this year's workshop includes papers that will broaden our horizons and enrich our teaching.

We hope that the readers of this volume will join the conference organizers, authors and participants for future workshops and conferences on information security education.

There are many who made this workshop possible, but special thanks goes to National Science Foundation whose support of the workshop allowed us to introduce many newcomers to the art and challenges of information security education. Some of the material in the proceedings and certainly the existence of this year's workshop itself is based upon support by the National Science Foundation under Grant No. 0210762. Of course, any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

We are grateful for the work of the members of the program committee who volunteered their time. The organizing committee has provided us with the support in the complexities of conference mechanics. This workshop would not have been possible without the help of those whose quiet behind the scenes efforts make the workshop run smoothly. We would especially like to thank Naomi Falby for her invaluable support in helping with the organization of the conference. We are grateful to Matthew Rose whose attention to detail has resulted in this well crafted proceedings. Our thanks extend to David Riebandt and several of our Naval Postgraduate School students, who have provided support during the conference.

July 2004

Cynthia Irvine

SECURITY EDUCATION FOR BUSINESS

William Hugh Murray, CISSP

Naval Postgraduate School

Colleges and universities educate their graduates for three primary markets. American schools do nothing quite so well as they prepare their graduates for more schooling. Government seems well satisfied with the quality, if not the quantity of current graduates. This presentation will focus on what business wants, if not expects, from your product in the hope that doing so will improve the chances that we get it.

It will assert that business prefers candidates that are educated, as opposed to trained. Business needs team players, not prima donnas. It prefers trained, disciplined, rigorous thinkers to those who know a lot, candidates who are skilled as well as knowledgeable. It needs candidates who, not only can be expected to behave ethically but who know enough about ethical analysis to identify the right behavior in novel situations. While it values specialists and is often willing to pay a premium for them, most of its candidates must have a liberal education.

Business trains and educates, has always done so. However, we want to spend our time and money teaching those things that only we can teach. We do not want to teach the fundamentals and we do not want to fix your failures. We expect that college graduates be able to present their thoughts in coherent written and spoken English. We expect that they understand such fundamental logical concepts as Occam's Razor, necessary and sufficient, and proper, ordered, and complete lists.

We understand that people learn right from wrong in sand pile; we do not expect colleges and universities to teach their students how to behave well or to remediate the failures of kindergarten. However, we are disappointed, not to say frightened, when college graduates do not recognize, much less are able to reason about, such fundamental ethical concepts as the rule of law, the sanctity of contract, and the categorical imperative.

No other generation in history has been able to lavish twelve, sixteen, or even twenty years on the education of its young, (nor been as little satisfied with the result). As recently as a hundred years ago, only a tiny elite could aspire to such an indulgence. For most that much time would have represented a third or even half of their life expectancy. Surely we are not investing so much of our national treasure in our young simply to produce workers for an increasingly automated industry, as our national policy makers would have us believe,. Surely education is about more than that. We can get labor from high schools, and skilled workers from community colleges and

technical schools. From colleges and universities w need citizens, not workers. We need people who have read enough history that they can choose their future, enough science that they can recognize truth, and enough art, music, and drama that they can appreciate and foster beauty.

We expect a college degree to mean more than that someone has spent the requisite number of hours in a classroom.

The presentation will defend these assertions in terms that the speaker hopes will be familiar to academics and to which he hopes they will be sympathetic.

TEACHING COMPUTER SECURITY TO UNDERGRADUATES

A Hands-On Approach

Rahul V. Tikekar

Southern Oregon University

Abstract:

Increasing awareness of the vulnerabilities of computer systems has led to the introduction of several programs in computer security. Many of these programs are meant to attract graduate students. Southern Oregon University has recently started a new undergraduate track in computer security and information assurance (CSIA). The curriculum is first of its kind in Oregon to be available at the undergraduate level. The CSIA curriculum is based on a set of core classes from computer science, mathematics, and inter-disciplinary topics. Laboratory and hands-on exercises are used extensively to augment the matter covered in class. This paper describes the laboratory and other hands-on exercises of one course in the CSIA curriculum.

Key words: CSIA curriculum, undergraduate curriculum, computer security education, lab exercises

1. INTRODUCTION

Strengthening our nation's defense against malicious IT attacks on computer systems has taken on an important role in the last few years. In this time period the Internet has grown from a static web "homepage" distribution system to a flourishing society that is engaged in transacting business all over the world. Our society and economy today are dependent on information technology and information infrastructure [1]. This dependence has introduced many security risks that have manifested in the form of attacks on computer systems.

With the ever-increasing number of threats to computer systems, employers will be looking for qualified professionals to keep their systems and information secure. It is therefore important for universities to incorporate computer and information systems security into their curriculum. The 'Federal Cyber Service: Scholarship for Service Program' [2] seeks to increase the number of qualified students entering the fields of information assurance and computer security and to increase the capacity of the United States higher education enterprise to continue to produce professionals in these fields.

SOU is a 5000-student, publicly funded, liberal arts university serving a rural population 300 miles from the nearest large city. The computer science department offers four undergraduate

tracks: computer science and programming, computer information systems, computer multi-media systems, and computer security and information assurance (CSIA). A masters degree is also offered that is aimed at preparing students for the current job market. A major department goal is to enhance its reputation and ability to recruit nationally in order to expand the pool of potential students. It is the hope of the department that the CSIA curriculum will attract students to the school and the department.

The complete course requirements and description of the CSIA curriculum can be found at the department's web site [3]. One of the focuses of the curriculum is to provide students with hands-on experience in real-world problems. This experience is provided through lab exercises, class projects, and capstone projects. In the summer of 2003, the author attended the Workshop on Education in Computer Security (WECS5) and the World Conference on Information Security Education (WISE3). This paper describes some of the lab and hands-on exercises, and projects used by the author – some a result of knowledge gained by the materials and papers presented at the workshop and conference.

The rest of the paper is organized as follows. Section 2 briefly describes three important courses in the curriculum and the lessons learned from WECS5 and WISE3. Section 3 describes the lab – the equipment and layout, and describes some of the exercises used; they are still evolving. Finally section 4 presents conclusions.

2. THREE IMPORTANT COURSES

The CSIA curriculum [3] is built around a three-course sequence: CS457 Computer Security I, CS458 Computer Security II, and CS459 Computer Security III.

The course Computer Security I covers the many facets of computer security and information assurance. It explores the security organization and infrastructure within an organization along with its policies, standards and procedures. Cryptographic protocols and algorithms (like DES, AES, RSA, Kerberos, digital signatures, etc.) are covered. This course helps lay the foundation of computer security and information assurance.

The course Computer Security II covers techniques and principles of design and configuration of secure workstations, servers and LANs. Topics like user and password management, system logging, intrusion detection techniques, etc. are presented. The course also covers the basics of virtual private networks, routers, firewalls and their implementation.

The course Computer Security III is a hands-on study of the threats to computer systems connected to the Internet. The overall objective is to understand how crackers find a system, find vulnerabilities in that system, and use a vulnerability to compromise the system, including the use of viruses. Various tools to attack and defend a system are studied. This is the course that the author teaches and the students spend a large percentage of the term in the lab. It is therefore necessary to design appropriate lab and hands-on exercises using tools that will help make this course an interesting experience.

2.1 Useful lessons from WECS5 and WISE3

The presentations from WECS5 and WISE3 provided very useful materials for those intending to offer courses in CSIA. In the case of the CSIA curriculum at SOU, many of the topics presented were already included in the CS457 and CS458 courses (e.g., cryptography, common criteria, firewalls, IDSs, etc.). This section lists those topics included in the course that the author teaches – CS459 Computer Security III. First, the topics from WECS5 included in the course are listed along with the approximate number of lecture and lab hours devoted to that topic. Many of these topics use the lab to gain hands-on experience. Detailed lab exercises are described in a later section.

a. Passwords and Password Cracking (2): Students are presented with the parameters of good password – use of special characters and longer lengths. These strengthen the passwords and make them less vulnerable to the cracking algorithms. A lab on using the password cracker, John the Ripper, helps illustrate the points.
b. Digital certificates and key exchange (1): Students are shown how a digital certificate facilitates in the exchange of keys to accomplish symmetric encryption. The case of a web browser and a secure server at an online store are taken to illustrate these concepts.
c. Covert Channels (1): Students are introduced to the technique of using a high level program to modulate a resource that a low level program can detect thereby circumventing the access rules and effecting a transfer of classified or other protected information. Students are required to write programs that conduct such "covert" communications.
d. Malicious Software and Attacks (4): Students are introduced to the world of viruses – their creation, release, and detection. A lab exercise on creating a simple malicious program is used to illustrate the concept.
e. Packet Sniffing and Spoofing (4): Students learn how packet sniffers are used to gather information passing between computers and how spoofing can help attackers gain access or even hijack entire sessions. IP spoofing, ARP spoofing, and DNS spoofing are discussed.
f. Steganography (2): Students learn how information can be hidden in other information, for example images and web sites. Student work on exercises using the OutGuess, Stegdetect, and Stegbreak [8].
g. Vulnerability Assessment and Penetration via the WWW (4): A series of lab exercises devoted toward understanding ways of exploiting vulnerabilities in Internet applications to gain access to a system. Some of the vulnerabilities covered are buffer overflows and WWW server and application-based attacks.

Next, the papers presented at WISE3 [4] that provided useful material for inclusion in the CS459 course are listed. Not all of these have been incorporated into the curriculum as yet but it is the author's strong desire to do so in the next year.

a. Teaching Network Security Through Live Exercises by G. Vigna. This paper presented an interesting tool that can be used in a class on computer security to test the students' understanding of the concepts and their being to use them in an exercise against other teams. Such an exercise immediately generates excitement and enthusiasm in students.
b. Design of a Laboratory for Information Security Education by V. Anantpadmanbhan, et al. This paper provides details on how a laboratory may be designed that will aid in the instruction of CSIA topics. They present some very interesting ideas for building a reconfigurable laboratory.
c. Information Security Fundamentals by P. Oscarson. This paper provides graphical representations to illustrate security concepts like threats, incidents, security mechanisms, vulnerability, and risk. Such a graphical tool can be very useful when explaining how the fundamental pillars of security – confidentiality, integrity, and availability, can become compromised.

3. LAB DESCRIPTION AND EXERCISES

3.1 The Computer Security and Network Lab

The computer security lab, whose figure is shown in [9], is built using PCs running Windows XP and Linux, and equipment donated by Cisco (firewalls, routers, and switches). The objective

behind the design of the lab was to create one that would model a real enterprise network – subnets, routers, firewalls, DNS server, Internet access, etc. would be a part of such a network. The lab therefore serves the needs of the CS459 course as well as the Unix systems administration, and introductory and advanced computer networking courses. The machines have both Windows and Linux operating systems installed on them. In addition to the equipment described, there are machines that serve classes on computer forensics and high performance computing.

Such an arrangement presents some challenges:

- Many of the classes have more students that computers and that means students have to share workstations.
- Some exercises (like password cracking, DNS cache setup, etc.) require system level access to the machines. Hence the system password needs to be constantly updated to prevent malicious use of the system. Also, many systems go into a "broken" state after the completion of certain exercises and so they have to be re-imaged after every lab.
- Some exercises run programs (like recompiling a kernel) that take more than the class time to complete. It is necessary to warn the students in advance of such an exercise so some of them can make time to stay back and finish the exercise.
- Staffing is necessary for the maintenance of the lab. Such staffing is currently provided through volunteer students eager for experience in system administration.

3.2 Lab Exercises and Projects

Lab exercises serve two objectives: to set aside one class period to give students a chance to experiment with the matter covered while in class, and to extend the exercise as an assignment. Here is a description of some of the lab exercises used in CS459 course, not already described before.

Using the whois databases to gather information about systems: The objective of this exercise is to show students how and where the .com, .org, .net, etc. domain names are registered. The exercise shows students how to query these databases, look for network administrators and find the IP addresses assigned to an organization. Attackers use these databases as a starting point for gaining access to systems by gathering information about them. Students realized how easy it was to get information about users and organizations on the Internet. Once this information is obtained, it is possible to run port-scanning tools on those systems to gather which ports are open. This scanning exercise was carried out on computers inside the lab.

Using password analysis tools like John the Ripper to identify bad passwords: Bad or default passwords are the basis of many methods used to gain access to a system. The objective of this exercise is to show students how easy it is to crack bad passwords and to motivate them to use better passwords. It also helps students wanting to be system administrators understand the importance of formulating policies on good passwords. The challenge is to use systems without real users and yet have a sufficient number of passwords as in an actual system. The program takes a very long time to run. There were no real accounts on the systems and so a few test accounts were created. The tool was able to crack the passwords of the test accounts in a relatively short time.

Using a packet sniffer (like ettercap, snort, etc.) to analyze network traffic: Attackers use packet sniffers to look at network traffic; this way they gather, among other things, userIDs and passwords as they are being passed between systems. Packet sniffers are also used for legitimate reasons to diagnose network problems. The objective of this exercise is to show students how computers exchange packets (e.g., TCP handshake), the structure of a packet and how information is encased in them. This assignment also helps in explaining spoofing and session hijacking concepts. Students realized how unsafe telnet and ftp utilities were when they saw their user names and passwords being transferred unencrypted over the network.

Recreating a virus: The objective of this exercise is to understand the anatomy of a virus attack. A simple virus (like Melissa) is created in the controlled environment and propagated. The students learn how viruses are written, how they are propagated via mediums like email, how they access system resources once in a system, and how to disinfect them. This exercise provides students with an excellent hands-on look at the way a virus works. The matter learned at WECS5 is augmented by material from the book on Viruses Revealed [5].

Web-based intrusions: Students are presented with a sample web site like the one given in the book on Web Hacking [6]. This exercise is an attempt to help students understand the vulnerabilities in web servers and applications. One of the exercises is modeled after a hands-on exercise at WECS5 in which the participants were able to deface a web site by exploiting vulnerability in the IIS web server. The exercises are used to introduce such concepts as buffer overflows, understanding web URLs and use of hidden variables, SQL injection attacks, Java remote executions, and session hijacking. Various small web applications are used to demonstrate techniques used to break into a web application.

Students also work on a project as a requirement for the course. The project requires students to perform research in two parts: known exploits and cyber laws. The first part requires students to lookup some recent examples of intrusions and viruses (in the case of this term, the Sasser worm had gained popularity) and then to document that exploit or virus – how it worked and what can be done to patch the vulnerability. Such an exercise will prepare students who will go on to become system administrators to handle such situations. The second part is an attempt to understand current and pending state and federal laws that address black hat activities and crimes.

In the last week of classes a contest is organized that has students capture one or more flags by breaking into a given target system. The class is divided into teams and the team that captures the most flags wins. This project is based on the CTF contest described in [7]. A target server is created that runs 6 services with known vulnerabilities. Since the students in the class are undergraduates they are told about the services and the vulnerabilities so that they may be able to actually perform the break-ins. Also, an entire day is given for this exercise.

4. CONCLUSIONS AND OBSERVATIONS

Designing a hands-on course that covers the important aspects of computer security is a challenging proposition and this paper has attempted to show one approach. In general, undergraduates require more handholding than graduate students. Hence a hands-on approach will require more instructor involvement. As a result some exercises will take longer than the class period to complete. Having an administrator to help with setting up the lab environment is crucial to the success of a hands-on approach. Most programs for the Unix platform are available as source distributions containing makefiles and configuration file that often need to be edited. Hence it is also important that students be familiar with downloading, configuring, building, and installing software.

Also, in the newer operating systems, many of the famous weaknesses have been strengthened. As a result recreating, implementing, or experimenting with those vulnerabilities will not succeed. For example, implementing a stack overflow as described in [10] will not work since current operating systems have implemented a stack guard. As another example, most new operating systems do not provide a telnet daemon. As a result, demonstrating session hijacking becomes difficult. As yet another example, some tools like Dsniff, will not install on new operating systems, as they need an older library that is no longer available. The conclusion to be drawn from these experiences is to use an older release of an operating system (e.g., RedHat Linux 6.2) to teach such a class.

Another issue that the author has faced in this class is bimodal students; there are almost always a few students who are very knowledgeable in this field and can overwhelm the other

students. In the past the author has given such students the option to test out of the course. On the upside student enthusiasm is very high in a course like this since it involves hands-on work with material that is very current and relevant. Many students use the knowledge to test their own or their relatives' systems, only to discover the many holes that are present in them.

5. REFERENCES

1. White house paper on securing cyberspace: http://www.whitehouse.gov/pcipb/cyberstrategy-draft.html
2. SFS program: http://www.ehr.nsf.gov/ehr/DUE/programs/sfs/
3. SOU's CSIA website: http://www.csia.sou.edu
4. C. Irvine and H. Armstrong (editors), Security Educational and Critical Infrastructures, Kluwer Academic Publishers, 2003
5. C.D. Harley et al. (authors), Viruses Revealed, McGraw-Hill Osborne, 2001
6. S. McClure et al. (authors), Web Hacking: Attacks and Defense, Addison-Wesley, 2002
7. Capture the Flag: http://www.cs.ucsb.edu/~vigna/CTF/
8. Steganography resources: http://www.outguess.org/
9. Computer network and security lab: http://www.priscilla.com/cis336/lab.htm
10. Aleph One (author), Smashing the Stack for Fun and Profit: http://www.phrack.org/show.php?p=49&a=14

A NOTE REGARDING COVERT CHANNELS

Timothy E. Levin and Paul C. Clark

Naval Postgraduate School

Abstract: This note presents an overview of some abstract concepts regarding covert channels. It discusses primary means of synchronization and illicit interference between subjects in a multilevel computing environment, and it describes a detailed laboratory exercise utilizing these abstractions.

Key words: security, covert channel, interference, multilevel, mandatory

1. INTRODUCTION

In a multilevel computing environment, a security policy is enforced which requires that low-sensitivity subjects (e.g., a process or task) should not observe high-sensitivity information (e.g., data, code, or activities of high-sensitivity subjects). The most intuitive interpretation of such a policy is a confidentiality policy, in which for example, subjects with a low clearance are not allowed access to highly classified data[1].

A multilevel system may enforce such a policy on all subjects under its control and all of the objects that it exports to those subjects (viz., objects to which an explicit reference is possible via a system interface). Such an enforcement mechanism is said to enforce *mandatory access control* (MAC) with respect to the *exported objects*.

Despite the successful enforcement of MAC, a *covert channel* exists in such a system when information can be passed from a high sensitivity *sender* subject to a low sensitivity *receiver* subject via an *internal* object (i.e., one that is not an exported object). This reflects a processing model in which all interactions between subjects occur through objects of some type, such as buffers, messages, registers and files.

Covert channels are normally conceived as a medium for a *series* of transmissions from high to low. Thus, for each transmission, the receiver has to know when to read. This is done through a *synchronization* mechanism. There also needs to be something – the internal object – that the sender can modify and the receiver can observe: this forms the *interference* mechanism of the channel, as shown in the Figure 1.

[1] A similar interpretation applies to *integrity* policies, e.g., wherein subjects with high integrity should not execute or observe low-integrity data or code.

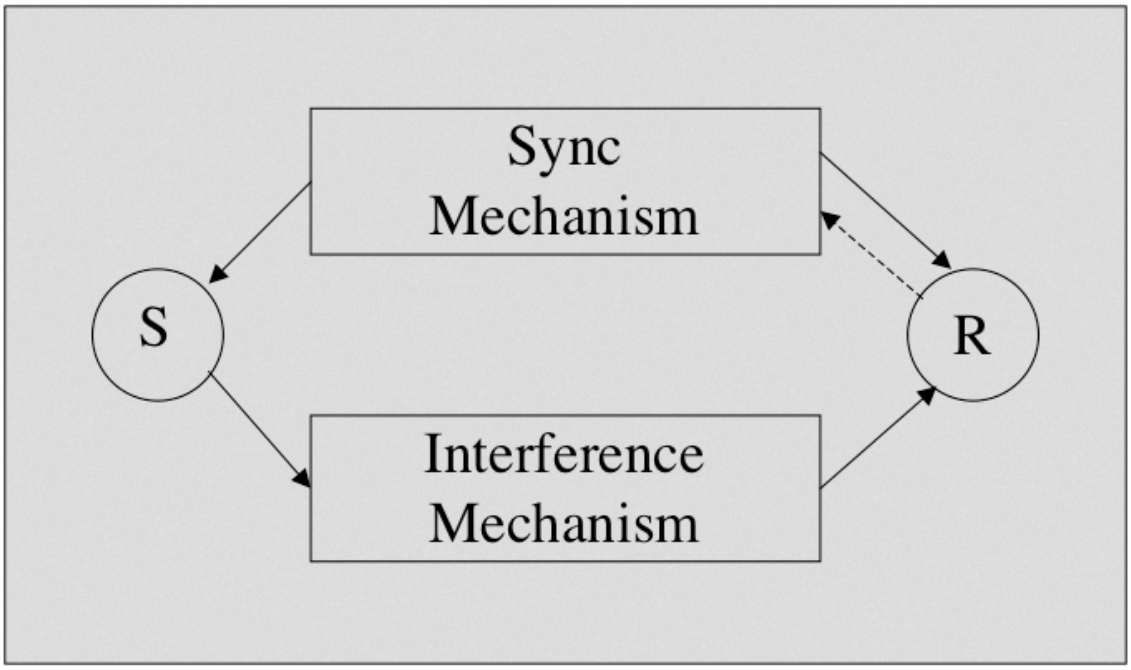

Figure 1. Abstract Covert Channel

2. SYNCHRONIZATION

An important distinction for a multilevel security system is that the sender (high) cannot send synchronization signals to the receiver (low) except through covert channels (if a receiver can *directly* read an external object that a sender writes to, it is considered a system flaw rather than a covert channel). The receiver, on the other hand, can send "I am ready" synchronization signals to the sender, in which case, the synchronization mechanism is a legitimate "channel" from low to high. Alternatively, they can synchronize via out-of-band (temporal) agreements relative to a "clock" mechanism that they both can read, such as, "you write on the odd time periods, and I will read on the even time periods. Synchronization can also be achieved through a mutually exclusive scheduling mechanism such as is commonly provided by an operating system.

In many descriptions of theoretical covert storage and timing channels, the sender and receiver synchronize by reading a clock or by depending on a scheduler to alternately activate them, because polling and the use of additional covert channels for synchronization may be less efficient for the purposes of optimal channel capacity estimation.

3. CLOCKS

If there exists a fluctuating variable, not modified by the sender, which the receiver consults to interpret or decode the data received, then that variable is effectively a "clock." A clock can have regular or irregular progression; and it can have different possible values, including a binary range. Any modulated (e.g., fluctuating or monotonically increasing or decreasing) variable can be used as a "clock."

4. INTERFERENCE

The sender *interferes* with the receiver through what we call a "control structure," such as a file lock or disk-full indicator for a *storage channel*, and a scheduler or disk arm location for a *timing channel*. In most storage channels, the receiver indirectly reads the interference information via an error message; for timing channels, the receiver *interprets* the interference (e.g., how long was my delay) via a clock. This is shown in the Table 1.

	Synchronization Mechanism	**Interference Mechanism**	**Interpretation Mechanism**
Storage Channel	Clock/Scheduler	Control Structure	Error Message
Timing Channel	Clock/Scheduler	Control Structure	Clock

Table 1. Covert Storage and Timing Channels

The sender's activity in a timing channel can be viewed abstractly as interfering with the receiver such that the clock variable, when read by the receiver, will be at the value that the sender wishes the receiver to perceive. This is true regardless of whether or not the clock is monotonically increasing by regular values. It is only required that the sender knows what effect its actions will have on the clock value perceived by the receiver.

For a priority scheduler based timing channel, the sender interferes by consuming a smaller or bigger time-slice. When the receiver is next scheduled (completing its wait for the CPU), it reads the "clock" to interpret the bit or bits transmitted – the clock is not the channel, as the transmission of information has already occurred in the *delay*. In another timing channel example, the control mechanism is an internal data structure (e.g., shared table or disk arm location), which is modified, added to, or deleted from by the sender to interfere with the receiver; the receiver indirectly accesses the control structure through (e.g.) a synchronous operation; after completing its wait for the operation, the receiver interprets the bits transmitted by reading the clock.

5. COVERT STORAGE CHANNEL EXERCISE

In this section we present a detailed description of a covert storage channel that is present in certain file system implementations. File system directories "contain" files, as well as a description of those files. When a file is created in a directory, the process creating the file abstractly modifies that directory. In a multilevel environment, then, normal secrecy rules require that the secrecy level of the process creating the file must be *equal* to the secrecy level of the directory. This presents some practical problems, such as maintaining publicly writable directories, like /tmp in Unix: how can processes at every secrecy level write to a directory if they all have to be at the level of the directory? To remedy this problem in Trusted Solaris[2], it supports the notion of a Multi Label Directory (MLD), sometimes referred to as a *deflection directory* in other multilevel operating systems.

An MLD is assigned the secrecy level of the process that creates it, but files may be created in the MLD by any process whose secrecy level is *greater than or equal to* the level of the MLD. The OS does this by creating hidden subdirectories at the level of the creating process, and creating the file there. For example, a process at the UNCLASS level can create *file1* in an MLD,

[2] It should be noted that this behavior exists in an older version of Trusted Solaris. The existence of this behavior has not been verified on the newer releases.

and a process at the SECRET level can create *file2* in the same MLD. When the directory is viewed from the UNCLASS level, only *file1* can be seen, and when viewed from the SECRET level, only *file2* can be seen. As desirable as this functionality is, it creates a covert channel.

The cause of the covert channel is traced to another desirable aspect of file system directories: we should not be able to delete a directory that still has files in it. A process will receive an error message if it attempts to delete a directory that is not empty. Given a low-level MLD containing a high-level file, a low-level process cannot see the high-level file, but if it tries to delete the MLD, it will receive an error message, and thus know that one or more high-level files exist in that directory. In other words, the covert channel exists because a high-level process can modify the state of the MLD, interfering with the ability of the low-level process to delete the directory.

At the Naval Postgraduate School, one of the lab exercises assigned in the *Introduction to Computer Security* course has the students take advantage of this covert channel to transfer data from a high secrecy process to a low secrecy process. In this exercise, the channel is initialized when a low-level process creates an MLD known by a high-level process. The two processes must also agree on the meaning of the signal. It is assumed that if the low-level process can delete the directory, then a bit value of 0 is being passed from high to low. If the low-level process cannot delete the directory, then a bit value of 1 is being passed. The high-level process puts something into the agreed upon MLD, or not, depending on the value of the bit being passed. There must also be some synchronization mechanism for knowing when the low-level process is ready to receive information, when it should try to delete the directory, and some mechanism for knowing when the full message has been transferred. Conveniently, this communication can also be arranged with other MLDs. In addition, it is possible to transfer more than one bit at a time by making use of multiple MLDs at a time.

Putting this all together, the following gives the steps that both sender and receiver use to transfer the contents of a file, eight bits (one byte) at a time, from high to low. We start by assuming the sender and receiver know the location of the ".backdoor" directory used to stage the activities.

5.1 Low-Level Receiver Process

1. Create all the MLDs used to transfer each byte of information, i.e., .backdoor/bit0, .backdoor/bit1….backdoor/bit7.
2. Create the MLD used by the high-level process to signal when all the data has been transferred: .backdoor/quit. Once this directory is created, it also signals the high-level process that the low-level process is ready to receive data.
3. Repeat the following:
    a. Create the .backdoor/signal MLD. When the high-level process detects that this directory exists, it is a signal that the low-level process is ready to receive another byte of information. The high-level process puts a file in this directory after it detects its presence as a signal to the low-level process that it is still transferring information.
    b. Sleep for two seconds to give the high-level process time to wake up and create a file in the signal directory.
    c. Try to delete the signal directory. Keep trying until successful, waiting one second between each attempt. Once it can be deleted, the high-level process is done transferring a byte of information.
    d. Try to delete the .backdoor/quit directory. If successful, the high-level process is communicating that there is no more data to send, and the loop must be exited.
    e. Extract the byte of information by trying to delete each bit directory. If a directory can be deleted, then the associated bit is a "0", otherwise it is a "1". Build a byte from this information.
    f. Recreate the bit directories that were deleted in the previous step.

   g. Save the byte.
4. Delete all the bit directories.

5.2 High-Level Sender Process

1. Keep trying to create a file in the .backdoor/quit MLD directory until successful. The .backdoor/quit directory does not exist until the low-level process creates it. This becomes a signal to the high-level process that it has initialized the MLD directories required to transfer a byte, and is otherwise ready to receive information.
2. Do the following until the contents of the file have been transferred:
   a. Delete any files placed in the eight bit directories.
   b. Request information about the .backdoor/signal directory until no error is returned. If an error is returned, go to sleep for one second. When the high-level process is able to obtain information about the directory, it means that the low-level process has created it, which means that the low-level process is ready to receive a byte of information.
   c. Create a file in the .backdoor/signal directory. This prevents the low-level process from deleting the directory, which is a signal to the low-level process that the high-level process is not done transferring information.
   d. Get the next byte in the file to transfer and determine which bits are 1's.
   e. Create files in the associated MLD directories for those bits that are 1's. For example, if the 6th bit is a 1, then a dummy file is created in the .backdoor/bit6 directory. This prevents the low-level process from deleting this directory.
   f. Delete the file created in the .backdoor/signal directory. This allows the low-level process to delete the directory, signaling that the high-level process has finished transferring a byte of information.
   g. Sleep for two seconds. This allows the low-level process to wake up and delete the .backdoor/signal directory, signaling the high-level process that it is not done retrieving the information.
3. Delete any dummy files placed in the eight bit directories.
4. Delete the file in the .backdoor/quit directory. This allows the low-level process to delete the directory, signaling that there is no more data to transmit.

6. SUMMARY

Covert channel exploitation scenarios can vary widely, yet, most utilize the same abstract mechanisms to illicitly transfer information from a high sensitivity subject to a low sensitivity subject: synchronization and interference. We have provided concise descriptions of these abstractions, and used them to differentiate covert timing and storage channels, as well as to describe a concrete and detailed laboratory exercise.

CAPTURE-THE-FLAG: LEARNING COMPUTER SECURITY UNDER FIRE

LCDR Chris Eagle, and John L. Clark

Naval Postgraduate School

Abstract: In this paper, we describe the Capture-the-Flag (CTF) activity and argue that it contributes to a necessary component of the computer security curriculum. This component is the study of software vulnerability investigation. It is currently not properly emphasized in this curriculum. We discuss reasons for this situation and we go on to describe how CTF can be useful for educating students within this focus. CTF helps develop those computer security skills that enable students to identify new vulnerabilities before those with malicious intent find them. It also helps them to hone the core computer security skills.

Key words: "capture the flag"

1. INTRODUCTION

Proactive vulnerability analysis on existing systems is lacking from the modern computer security curriculum. Students do not learn how to locate and fix design, configuration, and application flaws in existing systems. Exercises such as Capture-the-Flag (CTF) teach these skills. As such, CTF fills a critical void in Information Security education.

Our paper is organized as follows. We argue that the vulnerability analysis education problem exists. We describe the current computer security educational approach in the section titled "The InfoSec Educational Environment". There we explore the underlying reasons for the deficiency in this approach and we go on to link these reasons for the educational disconnect with the current approach. We then go on to describe CTF in the section titled "CTF: A Lab to Fill the Gap". There we describe the setup for a CTF exercise and explore why it helps to train students to discover flaws in systems. We summarize our conclusions in the section titled "Conclusions".

We propose that CTF helps to teach those skills that we find to be lacking. CTF fills this niche naturally by providing a safe parallel for the experiences of crackers in the wild. The specifics of a CTF exercise may lead to a general approach to teaching these skills. We focus on these specifics here because they serve as an excellent starting point for two efforts. First, CTF provides immediate educational value. Computer Security departments can participate in CTF exercises to teach a set of valuable skills. In addition, educational researchers can use CTF as an aide and basis to understanding how best to convey these skills.

2. THE INFOSEC EDUCATIONAL ENVIRONMENT

There are a series of challenges that a student of Information Security faces. Educational programs structure themselves in order to meet these challenges. Curricula are also developed to help meet the expectations of potential employers. These expectations present themselves in the course of the day-to-day operation of Information Technology (IT) systems.

Organizations concerned with the security of their IT generally break it down along two lines. The security of an organization's IT flows from that organization's policies, which defines the organization's approach to the problem. The application of the policy is colored, however, by the timing concerns involved. First, these organizations are interested in being able to protect systems that they already have in place. They must do something to deal with known vulnerabilities, or any threats to their operation will have known avenues for doing them harm. Second, they are interested in planning for systems with more capabilities in the future that will better support their mission.

Information Security education has grown to support these concerns. These organizational goals tend to divide a student's education into two categories. We will call these the *protectionist* and *constructionist* approaches to computer security. The protectionist approach focuses on developing students' capabilities to protect existing systems from known vulnerabilities, given known risks. The constructionist approach focuses on developing students' capabilities to build new systems that are free from vulnerabilities to a high degree of assurance.

In his paper *Training the Cyber Warrior*, J.D. Fulp describes this divide using the terms *cyber tacticians* and *cyber strategists*, which correspond to our concepts of protectionists and constructionists, respectively:

> Cyber tacticians would focus on reducing the risk of existing fielded systems primarily through the application of appropriate safeguards … . Cyber strategists would focus on reducing the risk of future systems primarily through the application of structured and formal system design techniques that reduce system vulnerabilities [Ful03].

Teachers use lecture sessions and laboratory work to convey information and experience to their students. Both of these tools tend to support one or the other of the protectionist or constructionist approaches. To illustrate this, we will refer to some generic courses. There are specific counterparts at the Naval Postgraduate School for many of these.

Lecture sections are direct and in Information Security, as elsewhere, are used to convey important baseline facts. Even when there is a lab involved, it typically builds on prerequisite material presented in a lecture. Information Security lectures can cover understanding policy, grasping current threats, and learning how technology (including networking) works. Such lectures are clearly protectionist in nature. Other lectures can discuss the meaning of security and a robust process for developing secure systems. These lectures can include formal modeling and techniques, design approaches, and implementation standards. These are clearly constructionist.

Laboratory exercises (labs) are a more interactive teaching tool. They have the benefit that they can provide experience in a subject. They are often more appealing to students than other types of work. It is difficult, however, to convey abstractions and theories through labs, as complex material can be challenging to model in a lab. Many of the labs in existence parallel the class lectures described previously. Teachers have also experimented with more comprehensive labs that exercise a variety of skills and the coordination of those skills. A Cyberdefense Exercise (CDX) is a lab that is focused on network security from a defense point of view. The converse of this is a Red Team Exercise, which attempts to take advantage of known vulnerabilities to test an installation's defenses. Labs such as these are protectionist in their approach. They encourage using known processes to achieve some well-defined goal using established technology and procedures.

For example, in his paper *Teaching Network Security Through Live Exercises*, Giovanni Vigna gives an overview of several configurations of interactive teaching techniques. One of the experiments that he describes is a lab called Capture The Flag. As Vigna describes it, though, this form of CTF exercise continues to contribute to the protectionist educational approach.

> The team's goal was not to prevent the other team from breaking into the host. Instead, the priority was to detect the attacks of the opponents. In addition, each team had to attack the other team's hosts and retrieve the flags for each of the attacked hosts [Vig03].

Here, the priority is learning about intrusion detection. Intrusion detection is a tool that is helpful for identifying when attacks may be taking place on IT resources. It is often taught to students as a component of a network security framework, and as such it is part of the protectionist educational focus.

In contrast to such labs, there are other labs that provide interactive experience to students who are developing systems. These labs include training using the development tools. They can also expose students to experimentation with computing system components that can help students reason about future systems. These labs are clearly constructionist in their focus.

There are important skills that are overlooked in these approaches to teaching Information Security. The protectionist focus has students learn what policies need to be enforced, and how to enforce them. With this focus, students harden systems and react to problems based upon known vulnerabilities. Once these students become practitioners, they must keep up to date in order to keep their systems up to date. Looking to the future, the constructionist focus has students learn the design skills needed to put together systems that are robust in the face of threats. The constructionists incorporate policy from the beginning to ensure that systems conform to policy. The nature of vulnerabilities is a vital input to both of these areas. Who is trained to discover new vulnerabilities?

3. CTF: A LAB TO FILL THE GAP

We propose that Capture-the-Flag serves as an example of the type of material that needs to be included in Information Security education. Capture-the-Flag is a team-based sport that is essentially an exercise in controlled, time-sensitive system subversion (also known as cracking). Aside from a set of artificial goals that give the sport a measuring stick and a set of artificial boundaries that keep the sport contained, participants have an extraordinary amount of freedom. This freedom motivates students to experiment and forces those students to hone skills that are not normally covered in a standard Information Security curriculum. CTF is one component of an educational focus that is currently missing from institutional Information Security education.

One of the potential drawbacks to CTF is the amount of setup required. CTF is based on the concept of running one's own—and subverting others'—services. A service is any application that can be utilized remotely over a network. Many services, for example, make use of a common web server. There are a plethora of potential services that can be used. In fact, many CTF configurations have tried to mix some common services with some that are more arcane.

The CTF setup introduces the concept of a flag in order to monitor whether each of a particular team's services is available, and if so, who controls it. If a service is controlled by a team other than the team who owns that service, then it has been remotely compromised. A flag is some small string of data that identifies a particular team. In order to assign scores to teams, service flags are cryptographically "rotated" in an unpredictable way in order to monitor the duration for which a service is controlled continuously by a given team. There must be an automated way to perform this scoring. This "scoring server" accesses each team's services in much the same way that the participating teams access each others' services, but it may also have additional privileged access mechanisms for performing flag rotation that must be communicated

to the teams in advance. The development of the scoring server is the primary source of the complexity in the preparation for the exercise.

Game play is also quite complex, although given a correct setup it is straightforward. Each team is given space within the network topology. Each team is also given media, typically optical, such as a CD or DVD, containing working images of the systems sufficient for running the services. How the team organizes itself to satisfy the exercise's requirements is left entirely to that team.

The most important characteristic of a CTF exercise is its focus on the 0-day exploit. A 0-day exploit is a software system vulnerability that has not been previously disclosed. 0-day exploits include both vulnerabilities that have not yet been discovered as well as those about which certain groups may know but choose not to reveal. In contrast to Vigna's approach to CTF exercises described earlier, the ones in which we have been involved require the development of new exploits. This is a CTF exercise's primary benefit. In order to be competitive, teams must harden their systems as much as possible. In turn, this means that other teams must find new, previously unknown and hence innovative ways to undermine their competitors' systems.

We have broken down the important educational targets for developing 0-day exploits into eight areas. These areas are:

1. Consistently secure programming practices
2. Compiler theory
3. Assembly language
4. Operating system theory
5. Reverse engineering theory
6. Networking and practical protocol analysis
7. Exploit methodology, and
8. Ethics and disclosure

Many of these topics are traditional computer science areas of study. In an Information Security educational environment, these topics would all be taught with a concentration on learning where flaws could exist and discovering where flaws actually exist. These skills border on both protectionist and constructionist domains, but they are largely overlooked in modern Information Security curricula.

While the 0-day exploit is of paramount importance to a CTF exercise, these exercises develop a wide assortment of additional skills. Building up a picture of how the exercise has been constructed and finding targets to analyze develops computer forensics skills. Defending the systems requires a combination of network security and system administration skills. This defense must be responsive to new attacks, and it requires extensive knowledge of a wide array of system components, including various (and potentially arcane) operating systems. Participating effectively in a CTF exercise requires preparation. This preparation is targeted towards enabling the team to cooperate. It requires some technical work, such as network engineering, but it also requires proper team management. CTF is a thorough information security exercise. It integrates well with other Information Security educational targets and so it can be used effectively to train Cybersecurity professionals.

4. CONCLUSIONS

In response to the IT environment and the security needs of that environment, an educational program of protection and construction has developed. We suggest that these categories are very useful educational foci, but that they require additional support. We need to be training people to act like crackers and find new vulnerabilities in existing systems. Exercises such as Capture-the-Flag develop these needed skills in students.

We conclude that we need a third focus for students. We will call this focus the *destructionist* focus; its primary goal is to learn how to break software and computer systems and thereby expose the vulnerabilities in those systems. Destructionists complement both protectionists and constructionists. Destructionists work to uncover new vulnerabilities. This information naturally supports the protectionists, who specifically work to counter known vulnerabilities. Destructionists also work to expose new models of risk and to better understand the weaknesses in system composition. This information is useful to the constructionists in that it will provide them with a better foundation on which to base their design decisions.

In our observation, CTF has helped students understand the security relevancy of system details in a way not previously covered. Students participating in our formulation of a CTF exercise are forced to deeply and thoroughly inspect and understand the operating environment of IT systems. They must then be able to assemble this knowledge into a viable threat. Our description of the CTF architecture gives students almost free reign in their manipulation of the game systems. Students are forced to take everything into consideration as a possible avenue of attack.

There is still a good deal of work that needs to be done to incorporate CTF neatly into an educational environment. In our experience, evaluating students based upon their participating in one of these exercises can be challenging. Individual students are likely to have very different aptitudes, and a CTF exercise tends to draw out specific skill sets. A CTF exercise is unavoidably a team exercise. As a result, we have typically used CTF exercises as extra-curricular activities, although this is also due to the fact that we are still working to integrate CTF-related material into the curriculum. This has the side benefit that as an extra-curricular activity, CTF provides an avenue for students to interact with the hacker community.

5. REFERENCES

[Ful03] Fulp, J.D. Training the Cyber Warrior. Security Education and Critical Infrastructures; June 26-28, 2003; Monterey, California, USA; The International Federation for Information Processing. Kluwer Academic Publishers, Boston, Massachusetts, USA. 2003.

[Vig03] Vigna, Giovanni. Teaching Network Security Through Live Exercises. Security Education and Critical Infrastructures; June 26-28, 2003; Monterey, California, USA; The International Federation for Information Processing. Kluwer Academic Publishers, Boston, Massachusetts, USA. 2003.

SECURITY AS A COMPONENT OF A COMPUTER SCIENCE CURRICULUM AT AT LIBERAL ARTS COLLEGE

Everett L. Bull, Jr.

Pomona College

Abstract: We present the experiences of integrating topics in computer security into an undergraduate liberal arts computer science curriculum.

Key words:

Pomona College is a small, highly-selective, residential, liberal arts college located in southern California. Its 1500 undergraduate students major in traditional academic disciplines in the humanities, natural sciences, and social sciences. The computer science major prepares its students for a wide range of careers by providing them with the fundamental principles, tools, and concepts of the discipline.

We are a small program and cannot easily add new courses, although a security elective is under consideration. On the other hand, a benefit of our size is that the linkages between courses are flexible, and it is easy to add, adapt, and rearrange topics within existing courses. Customization is the norm among the faculty members.

Below, we give some examples of such customization by showing how computer security was integrated into existing courses at Pomona College in 2003–2004. Much of the material came from the Fifth Workshop on Education in Computer Security, held in June 2003 at the Naval Postgraduate School in Monterey, California

Computer security is an effective vehicle for conveying important ideas in computer science. First, security is of current interest. Students read about it and care about it. They encounter related material in courses on constitutional law, economics, and media studies. They feel the immediate effect when the network in the residence halls is the victim of an attack or when local policies affect their abilities to share files and communicate via instant messaging. Second, security cuts across the discipline of computer science. It touches on algorithms and complexity, architecture, programming languages, operating systems, and networks. Finally, security is accessible. Its basic ideas can be understood by first-year students, who can then build up their knowledge throughout their undergraduate careers.

1. CURRICULAR COMPONENTS

During the academic year 2003–2004, we introduced computer security into four places in the curriculum: our second course in computer science, a senior seminar, an intermediate course on computer systems, and a senior project.

1.1 Computer Science 52, *Fundamentals of Computer Science*

The second course in our major sequence is an attempt to give the students a broad introduction to computer science. Although it is programming-intensive, it is not *about* programming. It is about science. We emphasize recursion and correctness through the use of the functional paradigm as expressed in the language ML.

One of the first assignments was an exercise in list processing: The students were to write an "infinite precision" integer package. Later in the course, they used the package to implement RSA encryption. The students posted public keys and encrypted messages for others to decrypt. (To get things started, the author posted an encrypted passage from Dostoyevsky that contained the phrases, "I am a sick man. … I believe my liver is diseased." Very shortly into the assignment, the author received an electronic mail message from one of the students inquiring about his health.) Students learned first–hand the relationship between complexity theory and cryptography. In particular, those who did not follow instructions and implemented exponentiation as iterated multiplication learned a lot about the difference between linear and exponential time.

In the section on data representation, we briefly discussed graphics and saw a demonstration of steganography. A brief presentation had been developed by the author for another purpose, a presentation to the local senior citizens computer club. There were a series of images in which short stories by Edgar Allen Poe were hidden in a photograph of the author's dog. Even though the students knew what to expect, they were satisfied to see it so clearly. With Poe occupying only one bit per byte, the image of the dog was indistinguishable from the original. At three bits per byte, it was only slightly fuzzy. At six bits, it was Warhol-esque. The surprise was that the dog was still distinguishable even with seven bits of each byte allocated to Poe.

One of the later parts of the course was devoted to an introduction of assembly language and computer organization. We studied the stack discipline for subprogram calls and talked about buffer-overflow attacks. Although we did not go into the mechanics in great detail, we did discuss how strongly-typed languages like Java and ML can reduce the possibility of stack smashing.

We spent about seven one-hour classes on security-related material.

1.2 Computer Science 190, *Senior Seminar*

The fall senior seminar is required of all computer science majors. Faculty members select two topics and collect readings on them. Students read the papers and take turns summarizing and leading discussions.

In the fall of 2003, we chose the topics of Evolutionary Computing and Computer Security. The students found the security readings challenging. They acquired a firmer grasp of basic principles and gained a new vocabulary with which to discuss security issues.

A list of the computer security readings appears in the appendix. We spent a total of four two-hour meetings on security in the fall semester.

1.3 Computer science 105, *Computer Systems*

Our course uses the recent text *Computer Systems: A Programmer's Perspective,* by Bryant and O'Hallaron (Prentice Hall, 2003). It uses the question "How does a program execute?" to expose architecture and instruction sets, assembly language, compilers, operating systems, and network concepts. Two exercises, due to Bryant and O'Hallaron and not the present author, were closely related to security.

One assignment was an exercise in reverse engineering. The students were given an executable program called a "bomb" and used debuggers and decompilers to find the input strings which would prevent the program from "exploding." The other assignment had a similar theme, but this time the students launched successful buffer attacks to prevent explosions.

We spent approximately five seventy-five minute class meetings on this material in the spring semester.

1.4 Computer Science 191, *Senior Project*

Each graduating senior is required to carry out a project of his or her own design. One senior, Adam Carasso, was interested in file systems, specifically the secure deletion of data. His initial idea was to achieve *complete* erasure of files, but preliminary research convinced him that it was impossible and that further work would require a magnetic force microscope, so he shifted to encrypted file systems. His goal was to design and implement file system for Linux based on public key encryption. There was not sufficient time for implementation, but he did complete a design of a hybrid system.

Not surprisingly, key distribution and management was a central concern. Efficiency of public key systems was also an issue. To mitigate the performance penalty, Adam chose to encrypt data with a symmetric key algorithm, using the public key only to encrypt the symmetric key. The encrypted symmetric key was stored in a special data structure in the file system. He was therefore able to preserve random access to files. Also, vulnerability was reduced because only symmetric keys are written to memory, and the loss of one such key would reveal only a small portion of the data. The private key, whose secrecy is more critical, is stored on a smart card and never written to memory.

Adam worked independently throughout the spring semester; neither he nor anyone else knows how many hours he spent.

2. OTHER OPPORTUNITIES

In the courses just described, we took advantage of opportunities of the moment. In another place or another time, we would see a different set of possibilities. For example, there is a place (one could easily argue it is an *essential* place) for formal systems in our sophomore-level course entitled *Computation and Logic* or in *Large-scale Software Development.* Similarly, one might expect to see a discussion of language safety and type safety in *Programming Languages.* In the *Computer Systems* course mentioned earlier, it would be natural to include material on network security.

The point is not to make security the theme of every course in the major, nor is it to replace a focused, in-depth course on computer security. Rather, every undergraduate computer science major should that questions about security and privacy are integral to the discipline and can be found in nearly every subfield. Our examples are meant to illustrate that there are many "teachable moments" which an instructor may leverage.

Like computer security, material on ethics and professionalism may be integrated into the mainstream courses. The two belong together. It would be wrong to teach techniques of reverse

engineering without also mentioning intellectual property, to teach network snooping without also discussing privacy, or to teach stack smashing without also delineating social and legal responsibilities. The curriculum guidelines of professional organizations all recommend that there be consciously-designed components of society, morals, and law. We believe that some, but perhaps not all, of that material ought to be woven into central courses and not relegated to separate and easily-ignored modules.

3. CONCLUSIONS

Our experience shows that it is possible to introduce topics in computer security into standard, existing courses. Computer security can motivate and illustrate the primary topics in a course, just as the central subject matter can illuminate security. The cost, in terms of class time, is minimal because security examples are often direct substitutes for other kinds of illustrations. Our approach is not a substitute for an advanced security elective in the undergraduate curriculum; it is instead a way to reach a larger group of students—not just those who choose a specialized course.

Any introduction to computer security at the undergraduate level should be grounded in the *principles* of computer science. Current events and excitement about hacking techniques can provide energy and motivation, but they should not overwhelm a class. For the sake of both computer science and computer security, we want our students to come away with a lasting understanding of the fundamental concepts of the discipline.

APPENDIX. SENIOR SEMINAR READING LIST

The papers chosen for the senior seminar are not necessarily the "best" or "most influential." We select papers that represent a wide variety of topics, that come from different periods in the development of computer science, and that are written for various purposes. The final list includes some true classics, a few popular articles, and several narrowly focused research papers. Part of the exercise is to develop the student's ability to read critically and to discriminate among the papers. Accordingly, our list should not be taken as a recommended or comprehensive set of readings.

For the security section in the fall of 2003, the seminar participants read and discussed the following papers:

Thompson, "Reflections on Trusting Trust," *Communications of the ACM,* Volume 27 (8), August 1984, pp. 761–763.

Lampson, *Computer Security in the Real World*, Marshall D. Abrams Invited Essay, presented at the Annual Computer Security Applications Conference, 2000. A revised version with the same title appears in *Computer,* Volume 37 (6), June 2004, 37–46.

Ellison and Schneier, "Ten Risks of PKI: What You're not Being Told about Public Key Infrastructure," *Computer Security Journal,* Volume 16 (1), 2000.

Anderson, "Why Cryptosystems Fail," *Proceedings of the First Conference on Computer and Communication Security*, ACM, 1993.

Borisov, Goldberg, and Wagner, "Intercepting Mobile Communications: The Insecurity of 802.11," *Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking,* ACM, 2001.

Lampson, "A Note on the Confinement Problem," *Communications of the ACM,* Volume 16 (10), October 1973, pp. 613–615.

Gong, Mueller, Prafullchandra, and Schemers, "Going Beyond the Sandbox: An Overview of the New Security Architecture in the Java Development Kit 1.2," *Proceedings of the USENIX Symposium on Internet Technologies and Systems,* Monterey, California, December 1997.

Lampson, "Protection," *Proceedings of the Fifth Princeton Conference on Information Sciences and Systems,* Princeton, 1971. Reprinted in *ACM Operating Systems Review,* Volume 8 (1), January 1974.

Schneider, Morrisett, and Harper, "A Language-Based Approach to Security," in *Informatics—10 Years Back, 10 Years Ahead,* Lecture Notes in Computer Science, Volume 2000, Springer-Verlag.

These papers were included in the readings, but there was not time for discussion:

Lamport, "Password Authentication with Insecure Communication," *Communications of the ACM,* Volume 24 (11), November 1981, pp. 770–772.

Bellovin, "Security Problems in the TCP/IP Protocol Suite," *Computer Communication Review,* Volume 19 (2), April 1989, pp. 32–48.

Schneider, "Open Source in Security: Visiting the Bizarre," *Proceedings of the 2000 IEEE Symposium on Security and Privacy,* IEEE Computer Society, May 2000, pp. 126–127.

TEACHING CONTEXT IN INFORMATION SECURITY

Matt Bishop

University of California at Davis

Abstract: This paper discusses teaching the application of technical idea through non-technical discussions, especially the use of puzzles to engage students. After discussing the need for teaching students to evaluate contexts in which decisions about computer security must be made, we suggest the use of questions and scenarios drawn from political science, history, and other humanities to force students to apply or derive principles of computer security to unusual and unexpected situations. Our experience shows that students find the process enjoyable, stimulating, and effective.

Key words: environment, judgment, instruction, computer security

1. PROBLEM STATEMENT

Software engineers used standard software engineering processes to develop software. They installed it in a state-of-the-art touch-screen system, and spent great care in making the interface easy to use by even the most naïve user. They used encryption to protect the results being transmitted to a central server, and powerful systems to provide the initialization and computation of results, and enabled an electronic audit log so results could be validated. Then they applied an ISO 9000 process to ensure good quality control, and had their product certified by both US Federal and state testing laboratories. Last-minute patches and modifications were made to ensure the systems worked. Yet, when deployed and used one day, the machines failed repeatedly. As a result, the company's right to sell its systems was withdrawn, and the company is being investigated for violating several laws. What went wrong?

The vendor sells electronic touch-screen voting systems (called DREs), and the server used in Clerk-Recorders' offices to total the votes. The problem that the vendor encountered was a failure to recognize the particular context in which DREs are used. The public, and the candidates, expect the systems to meet certain (nebulous) requirements, but three that people agree upon are: correctness of recording the votes and counting them (accuracy), anonymity of any individual's vote, and availability of the systems. Unlike standard systems, however, the DREs and the server are used only on Election Day. Failures of most types of software are annoying, but delays of an hour are not catastrophic. But in certain environments (notably medical and aviation), delays may be catastrophic, and vendors use high-assurance techniques to ensure that chances of failure are minimal—and even then, they usually provide backups to handle failure. If voting systems are not

available, or inaccurate, voters are disenfranchised, which—in a democratic or republican society—is unacceptable. The DRE vendor's error was in not realizing the social context in which its product would be used resembled the time-critical environment more than the ordinary-use environment.

The vendor's error is a common one. Consider the popular phrase, "securing cyberspace". The term "security" has too many meanings in this context. What are the requirements? They vary from industry to industry, indeed from organization to organization; the requirements for an academic institution, a company, a public charity, and a government agency all differ. The problem is that the techniques that exist, and are being developed, to secure systems are being deployed without adequate consideration of the context in which they will be used. "Are these methods appropriate and effective *for this particular environment*?" is a question heard all too infrequently.

A computer security course providing a basic education in various aspects of securing computers and information must also teach students when to use techniques, and—more importantly—how to analyze social, political, and cultural environments and contexts to determine whether a particular technique or technology is appropriate. The latter should not become all-consuming, to the detriment of teaching the technology and application of principles to computers, but it must be attempted in some measure.

Doing do has several benefits. First, teaching the non-technical aspects treats security as a holistic problem, rather than a purely technical one. This is reality. Mechanisms that are acceptable in some environments are not acceptable in others. For example, military environments can enforce procedural controls much more effectively than can academic environments. In the former, the chain of command provides a framework for stating and enforcing procedures. In the latter, questioning and autonomy are encouraged to a much greater degree. Procedures must be created and promulgated differently to be effective.

But not only mechanisms differ; policies do, too. Counting votes in an organization is different than counting votes in a public election, because the former can be re-run if there are irregularities. Worse, the policy may not be clear until *after* deployment, requiring both procedures and mechanisms to be adjusted accordingly. If 1% of the votes must be manually counted to verify any electronic vote counting system, does generating paper copies of the votes from the electronic equipment and counting those satisfy the statutory 1% requirement? This is a matter of law, not technology.

No purely technical treatment of computer security can produce competent security experts. Policy defines security, and the policy is given *ex cathedra*. Policies need to account for people. They must counter human foibles. They must describe security in a particular context or set of contexts that include environment, law, and expected user base. Hence, security is a human issue, and teaching it as a purely technological issue introduces gaps that technology cannot close.

The "best practices" rules gaining in popularity are good examples of this. Most describe how to secure a system. But many presume some definition of "secure" without stating it, and without explaining *why* the choices are made. If full log files cause overwriting, shutdown, or disable logging, the "best practice" rules should prescribe a choice appropriate to the goals of the system and its environment. If the log space is great enough so that the log cannot be filled between snapshots (and the log is cleared after each snapshot), then the setting is irrelevant. Otherwise, the policy must indicate which of the choices will give the greatest benefit (or do the least harm). People who have been exposed to the interaction of technical and non-technical matters will be able to make such judgments; those who have not will tend to follow the best practices without additional thought.

This judgment, this ability to apply principles and creativity in unusual or unexpected situations, is what computer security courses must encourage and teach. The key question is how to do so while keeping the students engaged, and supplementing the technical aspects of the instruction.

2. SOLUTION

To meet these needs, a computer security course can tie security principles into experience and practice by drawing examples from other disciplines. These examples should focus on five general areas.

First is teaching students to question assumptions. This technique helps one locate security flaws in a system, by uncovering gaps between the security mechanisms. However, it also uncovers assumptions about the environment. Returning to the electronic voting machine example, one assumption was that poll workers would hear a voting card being ejected, and thus be able to tell if a voter voted twice (because they would hear the second card being ejected too). But in practice, the noise at most precincts would mask the noise of the card being ejected. The assumption was not valid in the particular environment in which the machines would be used.

This suggests that all parts of a problem must be examined. This seems counter to the traditional "top-down" methods of analysis that are customary in computer science. In these methods, problems are decomposed into smaller problems, the smaller problems solved, and the solutions combined to solve the larger problem. If each smaller problem is solved *in the context of the larger problem*, then the traditional approach works. All too often, though, the smaller solutions ignore the context of the larger problem, and do not work properly. A classic example comes from warfare, in which one assumes that winning all the battles means winning the war. King Pyrrhus would disagree. After he beat the Persians in one of the innumerable battles between the Greeks and Persians, his army was so decimated that he said, "One more victory like this, and we are undone."

Examination of a problem as a whole sometimes leads to unexpected solutions. This offers the instructor the opportunity to emphasize the importance of looking at all aspects of the problem. A (possibly apocryphal) story of an attacker who repeatedly broke into computer systems makes this point. All sorts of technical measures were tried, but none succeeded. The defenders called the police, who tracked the perpetrator, a teen-ager, to another country that had no laws against this activity—so, naturally, when the police in that country were called, they declined to take action. The defenders were stymied, until a policeman had an unusual idea. He called the teen-ager's mother and told her what her son was doing. The attacks stopped immediately. The policeman's observation that this was a human problem, and his thinking that a primal human emotion (love of, or respect for, a parent) might solve the problem, was both astute and effective.

This emphasizes the need to consider human beings, both as individuals and as members of an organization. Security does not occur in a vacuum. Requiring users to authenticate themselves by providing urine specimens will not work in many societies because they violate customs of privacy. Similarly, expecting low-level employees to refuse instructions from a corporate vice-president who can fire them is quixotic at best. Security measures must take these inhibitions into account.

The most effective way to get students to understand these issues is to engage them by providing puzzles, and asking the students to brainstorm creative solutions. This has two benefits. First, it forces the students to think and speak up. This allows the instructor to guide the discussion to consider a variety of approaches without providing an answer. (As we shall see, some puzzles simply have no correct answer.) Second, the students often enjoy a short break from technical material, and a good puzzle will lead them to either discover some principle, or apply some principle, that also applies to technical material. In our experience, the students enjoy working with the puzzles, and sometimes suggest solutions or approaches—or complications—that the instructor had not considered. This also demonstrates the need for multiple viewpoints when considering security issues.

3. SELECTION OF PUZZLES

The two paramount rules for the selection of a puzzle are that the puzzle must engage the students, and the puzzle must illustrate some principle relevant to computer security. The instructor must know something about the students. If, for example, the students are from industry, one should concoct puzzles from business organizations and the world of commercial information technology. At a university, puzzles from the bureaucracy that students face are effective, as are puzzles drawn from other academic disciplines. For most students, current events can serve to supply puzzles, as can puzzles about break-ins and responses. People enjoy "war stories", and providing conflicts from a battlefield, a war, or politics, usually gets students very interested.

As to the second rule, the instructor can use the relationship between the virtual "cyber" world and the physical, "real" world. Principles of computer security derive from older principles. For example, the Principle of Least Privilege [Saltzer & Schroeder 1975] is a variant of the "need-to-know" principle so popular with governments and other organizations. The Principle of Separation of Privilege [Saltzer & Schroeder 1975] is a formalization of the idea of "defense in depth". Other parallels abound. The instructor can take advantage of this to illustrate the relationship between computer security and the human, organizational, environmental, and other constraints that students must consider.

Creating puzzles is straightforward, once one finds a topic. If the inspiration comes from a passage in a book or article, one can ask the students how the passage demonstrates a specific principle. A more open-ended method is to ask the students which principles the passage demonstrates, and why. A variant is to ask how to apply the contents of the passage to a computer system. The author prefers the open-ended approach for two reasons. First, if the discussion strays, the instructor can bring the discussion back to the points he or she wishes to make. Second, the students sometimes think of values or relationships that the instructor has not considered. The open-ended approach also encourages the students to speak more freely, as the problem is not so constrained as in the first approach. In the author's experience, encouraging creativity requires encouraging the students to express and consider ideas that sound crazy. Neils Bohr's comment that a colleague's idea was crazy, but not crazy enough to be true, is as reasonable in computer security as it is in quantum physics.

News stories and current events are also a fertile source for puzzles. Here the approach is slightly different. Rather than ask about general principles, the instructor can ask the students to put themselves in the position of the people involved in the incidents, or in the position of an analyst who is seeking to prevent or enable the actions in the incident, and say how they would act or what questions they would ask. The latter is particularly important. Students need to understand that security usually involves dealing with incomplete information, and part of what a good analyst does is asking questions. Taking this approach teaches the students how to analyze a problem, figure out what *additional* information would help them decide, and how to ask for it. When discussing fiction or historical incidents, this approach may fizzle. Either the students feel too remote from the events, or they can "look in the back of the book" to see the answers. But students are living in the times of current events and news stories, and easily relate to the problems.

4. EXAMPLES AND EXPERIENCES

What follows are some example puzzles in various areas of computer security. These were used in several undergraduate classes to spark thought and discussion among students, as well as to bring out points that the instructor wished to emphasize. We presented one puzzle at the

beginning of each class, and had the students discuss among themselves for 5 minutes or so; then, the class discussed their conclusions and ideas. The puzzles fell into four (broad) categories.

4.1 Questioning Assumptions

Saul Alinsky's books *Reveille for Radicals* and *Rules for Radicals* provided fertile ground for this category. Alinsky was a protégé of John Lewis, and delighted in organizing the poor and disenfranchised to force those in power to respond to their needs. His descriptions of tactics give insight into ways to attack systems, both political and computer.

An example passage is the following illustration of one of Alinsky's rules of tactics for an organizer:

> The third rule is: *Whenever possible go outside of the experience of the enemy*. Here you want to cause confusion, fear, and retreat.
>
> General William T. Sherman, whose name still causes a frenzied reaction throughout the South, provided a classic example of going outside the enemy's experience. Until Sherman, military tactics and strategies were based on standard patterns. All armies had fronts, rears, flanks, lines of communication, and lines of supply. Military campaigns were aimed at such standard objectives as rolling up the flanks of the enemy army or cutting the lines of supply or lines of communication, or moving around to attack from the rear. When Sherman cut loose on his famous March to the Sea, he had no front or rear lines of supplies or any other lines. He was on the loose and living on the land. The South, confronted with this new form of military invasion, reacted with confusion, panic, terror, and collapse. Sherman swept on to inevitable victory. It was the same tactic that, years later in the early days of World War II, the Nazi Panzer tank divisions emulated in their far-flung sweeps into enemy territory, as did our own General Patton with the American Third Armored Division.[3]

Those who attack computers act as Sherman did: they ask about the assumptions the defenders are making, and attack in ways that the defenders have not prepared for. Unless the defenders can handle situations they do not expect, they react the way the South did: confusion, panic, and collapse. Now consider security procedures for handling attacks. Applying the lesson from this passage to these procedures, students see that they must be prepared to depart from procedures as needed, and know how and when to do so. A second benefit is that the discussion can educate the students on why rigidity serves people ill, and flexibility well, in incident handling.

4.2 Holistic Thinking

Holistic thinking asks students to look at problems in context, rather than in narrow technical perspective. The following puzzle illustrates one approach to encouraging this mode of thinking:

> Microsoft spent February of last year teaching its programmers how to check their code for security vulnerabilities and how to introduce common security flaws. Yet many Microsoft programs still have security vulnerabilities. What problems do you think Microsoft encountered, and will encounter, in trying to find and clean up the vulnerabilities in its systems?

Initially, students brainstorm the technical problems that Microsoft faces. But those are relatively minor compared to the multiplicity of environments in which Microsoft systems are used. Vulnerabilities are defined in terms of the local policy, and Microsoft cannot build systems to satisfy all those policies. So Microsoft programs will continue to have vulnerabilities. Further,

[3] [1], pp. 127–128

Microsoft supports backwards compatibility on their systems. Fixing vulnerabilities may break this feature. What are the trade-offs?

The instructor can also point out the difference between security vulnerabilities and poor coding practices. Buffer overflows may indicate security vulnerabilities; they always indicate non-robust coding problems [Bishop & Frincke 2004].

4.3 Human and Organizational Problems

Sun Tzu's *The Art of War* and Niccolò Machiavelli's *The Prince* are excellent sources for problems of this type, although many news stories provide fodder as well. The point of these stories is that security must take into account people and organizations.

This passage from *The Prince* enables an instructor to illustrate how organizational problems affect security considerations:

> It can be put like this: the prince who is more afraid of his own people than of foreign interference should build fortresses; but the prince who fears foreign interference more than his own people should forget about them. The castle of Milan, built by Francesco Sforza, has caused and will cause more uprisings against the House of Sforza than any other source of disturbance. So the best fortress that exists is to avoid being hated by the people. If you have fortresses and yet the people hate you they will not save you; once the people have taken up arms they will not lack for outside help. In our own time, there is no instance of a fortress proving its worth to any ruler, except in the case of the countess of Forli, after her consort, count Girolamo, had been killed. In her case the fortress gave her a refuge against the assault of the populace, where she could wait for succor from Milan and then recover the state. Circumstances were such that the people could not obtain support from outside. But subsequently fortresses proved of little worth even to her, when Cesare Borgia attacked her and then her hostile subjects joined forces with the invader. So then as before it would have been safer for her to have avoided the enmity of the people than to have had fortresses. So all things considered, I commend those who erect fortresses and those who do not; and I censure anyone who, putting his trust in fortresses, does not mind if he is hated by the people.[4]

Technical courses rarely emphasize the importance of security officers obtaining and retaining the good will of the users and system administrators. Without that good will, the officers will spend more time dealing with recalcitrant and upset authorized users than they will with attacks from outsiders. With that good will, the officers will have many more people reporting suspicious problems, and system administrators who are receptive to adding, configuring, and applying security mechanisms. The instructor can use this to load into a discussion of organization techniques and processes to encourage this type of collaboration.

4.4 Thinking Out of the Box

Unusual problems demand creative solutions. The following story from *The Art of War* illustrates this point.

> If we do not wish to fight, we can prevent the enemy from engaging us even though the lines of encampment be merely traced out on the ground. All we need to do is to throw something odd and unaccountable in his way.
>
> Tu Mu relates a strategm of Chu-ko Liang, who in 149 B.C., when occupying Yang-p'ing and about to be attacked by Ssu-ma I, suddenly struck his colors, stopping the beating of the

[4] [Machiavelli], p. 69

> drums, and flung open the city gates, showing only a few men engaged in sweeping and sprinkling the ground. This unexpected proceeding had the intended effect; for Ssu-Ma I, suspecting an ambush, actually drew off his army and retreated.[5]

Asking the students to apply this to computer security draws interesting reactions. The key is to get students to think about how appearing to be weak, or unconcerned, can help improve security. After some discussion, this leads to the role of deception in computer security, a very fertile area—and one, in the author's experience, that the students enjoy.

5. CONCLUSION

Our experiences in using this technique have been uniformly good. In evaluations, students cite the "puzzle time" as a very enjoyable, educational aspect of the course. During the discussions, most of the students speak up—the most serious problem encountered is crowd control! This has the side benefit of encouraging students to ask questions during class, and makes clear to them that we encourage them to bring up ideas and misunderstandings even when the student thinks they are stupid or silly. Sometimes, those "stupid or silly" ideas suggest approaches and insights that would otherwise be overlooked. This applies both for the non-technical, judgment aspects of the course and for the technical aspects of the course.

Judgment is a critical facility for engineers and scientists. But the trend in teaching computer science, and other engineering and science disciplines, is towards mathematical, analytical rigor. Analytic understanding and rigor are essential to understanding these fields, but equally important is the ability to apply the techniques and technologies appropriately. These fields are sciences, but their application is also art. The art of computer security needs to be emphasized far more than it is in most courses. Using puzzles drawn from non-engineering disciplines helps this process immeasurably.

References

Saul Alinsky, *Rules for Radicals*, Random House, Inc., New York, NY (1972).

Matt Bishop and Deborah Frincke, "Teaching Robust Programming", *IEEE Security and Privacy* **2**(2) pp. 54–57 (Jan. 2004).

Niccolò Machiavelli, *The Prince*, Penguin Books, New York, NY (1995).

Jerome Saltzer and Michael Schroeder, "The Protection of Information in a Computer System", *Proceedings of the IEEE* **63**(9) pp. 1278–1308 (Sep. 1975).

Sun Tzu, *The Art of War*, Dell Publishing Co., New York, NY (1983).

[5] [Sun Tzu], pp. 27–28

TOPICS IN COMPUTER SECURITY

for the undergraduate student

Jim Griffin

Cabrillo College

Abstract: Cabrillo College has just adopted a new program in Computer Network and System Administration for the purpose of preparing students for industry certification in fields relating to Information Technology and for admission into higher education at four-year institutions. This report describes how Information Assurance topics, exercises and labs were integrated into the existing courses of this program. Specifically, examples in the five areas of *passwords*, *encryption*, *viruses*, *spoofing*, and *steganography* are given.

Key words: security, passwords, virus, encryption, spoofing, steganography

1. INTRODUCTION

With the goal of integrating Information Assurance concepts into existing Computer and Information Systems classes, four target classes were identified:

- An Introduction to UNIX/Linux
- UNIX/Linux Shell Programming
- Perl Programming in UNIX
- UNIX/Linux System Administration

Information Assurance topics from the areas of Authentication, Encryption and Threats were chosen to incorporate into these classes. The Introductory course was ideal for inclusion of a password security exercise, since students obtain accounts on campus computers in this class. The two programming classes were appropriate for exercises in handling viruses, spoofing, and steganography. The MD5 Message-Digest Algorithm allowed a way to bring an encryption topic into the account administration portion of the administration course. The exercises and labs developed for these three topics in Information Assurance will be outlined and described in the following sections.

AUTHENTICATION:

This exercise in password security addresses one of the student learner outcomes for the Introduction to UNIX/Linux class:

Access a UNIX computer in a secure manner

The exercise described below acts as both a learning activity in password security and authentication as well as a means of assessing the success of the outcome. The password cracking tool can be used to determine if students are selecting adequately secure passwords.

Students in this class are given accounts to a remote UNIX server for their class work. The accounts are initially created in a locked state with their passwords being the same as their login names. The accounts are activated on the first day, and the students are instructed on how to log in. When all students are logged in, the instructor runs the **John the Ripper** [1] program, demonstrating how quickly the passwords are *cracked*. Students are then instructed on how to change their password, and after doing so, the cracking program is run again. This serves as a discussion for the concept of good vs. bad passwords. The WECS5 material provided below and discussed in the **Lab 1** for CS3600 is used to instruct students how to select secure passwords that are memorable.

Table 1. The Brute Force Attack

Alphabet Size / Password Length	26	52	93
4	4.57×10^5	7.31×10^6	7.48×10^7
6	3.09×10^8	1.98×10^{10}	6.49×10^{11}
7	8.03×10^9	1.02×10^{12}	6.01×10^{13}
8	2.09×10^{11}	5.34×10^{13}	5.60×10^{15}
10	1.41×10^{14}	1.44×10^{17}	4.84×10^{19}

THREATS

Threats to computer security come in all shapes and sizes, but the two programming classes using the UNIX shell language and Perl lend themselves to handling a variety of software threats.

Viruses

One of the student learner outcomes for the course, UNIX/Linux Shell Programming is:

> *Format the output and process the input for a shell script using the UNIX commands of sed and awk*

The lab exercise described below serves as an excellent way to give the students practice in performing string manipulation and file processing within a shell script. At the same time, the student is discovering the principles of virus replication, detection and eradication.

I provided a shell-scripted virus that, when run, would infect all shell scripts in the current directory and subdirectories one level below the current directory. The text of the script is included below. The spreading nature of the virus was demonstrated to all students in class, and it was pointed out that the virus caused no damage other than replicating itself.

Students were then assigned the task of writing a scanner shell script that would identify any infected files, and, if invoked with a specified option, would fix the script by removing the virulent code. Issues that came up in this lab were ones relating to the importance of correctly identifying and isolating the undesirable code. We discuss false positive and false negative identifications.

```
#!/bin/bash
VIRUS=`sed -n "2,18p" < $0`
FILES=`find . -maxdepth 2 2>/dev/null`
for i in $FILES
do
set `file $i` > /dev/null 2>&1
```

```
        if [ "$3" = "shell" ]      # Infect only shell scripts
        then FILE=`echo $1 | sed "s/:$//"`
             HDR=`sed -n "2,18p" < $FILE`
             if [ "$HDR" != "$VIRUS" ] # Don't infect twice!
             then
                 ORIG=`tail +2 $FILE`  # Get all lines except first
                 echo "#!/bin/bash" > $FILE
                 echo "$VIRUS" >> $FILE
                 echo "$ORIG" >> $FILE
             fi
        fi
        done
        echo Achoooo, I\'ve got a cold.
        exit 0
```

1.1 Spoofing

Another student learner outcome for the UNIX/Linux Shell Programming course is:

> *Write shell scripts that interact with the user as well as read and write files on the system.*

The lab exercise described below gives students the opportunity to practice reading the keyboard and writing to the terminal screen in a variety of ways. At the same time the student learns how login spoofing programs can be used to capture a user's password. Not only is the student challenged in trying to reproduce the activity of a user login as accurately as possible, but practices in detecting spoofing attempts are learned.

Login Spoof Program

One way that system security is compromised is by one program pretending to be, (spoofing), another program. Spoofing the login program is one way to catch a user's password. System Administrators need to be aware of these types of programs and have procedures for mitigating against this kind of attack. To see what is involved in this kind of spoof, your task is to write a shell script program that mimicks a user logon session from the initial logon message through to the display of the shell prompt in the user's home directory. You will mail the captured password to yourself, and otherwise try not to let the user know the login was faked.

The purpose of this exercise is not to capture unsuspecting users' passwords, but to serve as a discussion as to what it takes to successfully spoof a login and how you as a system administrator can guard against this type of attack.

1.2 Steganography

A Perl Programming in UNIX course includes in its content: arrays, data transformation and the language's use within the World Wide Web. The instructor for this course incorporated as a final lab project, a lab on steganography. An image was hidden in the low-order bytes of a bitmapped file, (bmp image). The bmp image was used so students didn't also have to deal with compression and complex structures. A 24-bit color model was chosen so that students could process the input byte at a time. The low-order bits of pixels that did not carry hidden information were zeroed out.

Once again, this lab challenges the student to perform bit-wise operations on a data stream and to write output according to specified format; at the same time, it teaches an information security concept relevant in today's world.

2. ENCRYPTION

Although cryptography and encryption are not specific topics in the UNIX/Linux System Administration class, understanding the content and structure of system accounting files is. As a class demonstration I show how the MD5 digest algorithm converts a user password into a hashed digest. I utilized the algorithm obtained from RFC 1321[2] and modified the sample program to produce the output shown below:

```
p a s s w o r d
70617373776f726480000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000400000000000000000

0123456789abcdeffedcba9876543210       # Starting values for the digest
2fc215c612579bdffcb97531eca86420
2fc215c612579bdffcb9753169a674e3
2fc215c612579bdf43a6037f69a674e3
2fc215c6c28bb94e43a6037f69a674e3
4753f190c28bb94e43a6037f69a674e3
4753f190c28bb94e43a6037f5773f069
4753f190c28bb94eac38daf15773f069
4753f1906383a24fac38daf15773f069
d15facfc6383a24fac38daf15773f069
d15facfc6383a24fac38daf16b75829c
d15facfc6383a24fba5f7d836b75829c
d15facfc00d4b1fdba5f7d836b75829c
bc9ed05800d4b1fdba5f7d836b75829c
bc9ed05800d4b1fdba5f7d838bc96995
bc9ed05800d4b1fd1188ec9d8bc96995
bc9ed05884ab52891188ec9d8bc96995
786d4fb8c9c8bbe900e70d7b86afb209
e443006dac206673de56c6b27cb7bf4e
5f4dcc3b5aa765d61d8327deb882cf99         # final md5 digest
```

The student learns that this algorithm is irreversible and that UNIX passwords are not stored anywhere on the system. The administrator can, therefore, not recover a forgotten password, but can only set a new one.

3. CONCLUSION

The Computer and Information Systems department at Cabrillo College is actively pursuing the development of Information Assurance content into its new program. The effort represents a combination of folding particular topics and activities into existing courses, as well as developing new courses on specific IA principles

We have surveyed our students for CIS topics they would be interested in learning about. The results of that survey suggest that we develop half-semester courses on such topics as:

- Attacks and Countermeasures
- Intrusion Detection
- Computer Forensics

Although we have not done anything specifically to accommodate underrepresented groups in Computer Science, we have a separate program titled, *The Watsonville Digital Bridge Academy*[3], which is helping underrepresented groups to make a successful transition to college through such themes as support services, learning-to-learn, time management and self confidence.

The success of integrating IA concepts into our existing courses was manifested by an increase in student interest in performing the labs and exercises that related to real-world and sometimes "taboo" subjects. The relevancy of their work instilled a greater desire to succeed and was therefore motivating. In choosing a lab exercise for the students, it is important that the difficulty of the assignment match the objectives of the course as well as the capability of the students.

NOTES

1. John the Ripper is a password cracker, currently available for UNIX, DOS, WinNT/Win95. Its primary purpose is to detect weak UNIX passwords. It has been tested with Linux x86/Alpha/SPARC, FreeBSD x86, OpenBSD x86, Solaris2.x SPARC and x86, Digital UNIX, AIX, HP-UX, and IRIX. http://www.openwall.com/john/
2. Network Working Group , R. Rivest, Request for Comments: 1321
   MIT Laboratory for Computer Science and RSA Data Security, Inc. April 1992
3. Watsonville Digital Bridge Academy: http://www.cabrillo.cc.ca.us/~wdba

EXPRESSING AN INFORMATION SECURITY POLICY WITHIN A SECURITY SIMULATION GAME

Cynthia E. Irvine and Michael F. Thompson

Naval Postgraduate School

Abstract: The Center for the Information Systems Studies and Research (CISR) at the Naval Postgraduate School has established a broad program in computer and network security education. The program, founded on a core in traditional computer science, is extended by a progression of specialized courses and a broad set of information assurance research projects. A CISR objective has been improvement of information assurance education and training for the U.S. military and government. Pursuant to that objective, CISR is developing a computer simulation game, CyberCIEGE, to teach computer security principles. CyberCIEGE players construct computer networks and make choices affecting the ability of these networks and the game's virtual users to protect valuable assets from attack by both vandals and well-motivated professionals [1]. CyberCIEGE includes a language for expressing different security related scenarios. A central part of this language is an ability to express a variety of different information security policies.

Key words: Information Assurance, Security Policy, Simulation Game, Scenario Definition Language

1. INTRODUCTION

Computer and network security is a broad field with many subtle properties, not the least of which is its role as a negative requirement. In other words, the issue is often what does not happen, e.g., a secret asset should not be disclosed to an enemy. Educating students to appreciate the subtle elements of computer and network security is greatly enhanced by a tangible context in which cause and effect can be experienced. One means of providing this context is the use of labs containing networked computers and exercises illustrating specific attacks and defenses. Use of labs to provide context to lessons is limited to those students with access to the physical lab equipment. Also, students typically perceive a lab exercise as just that: an exercise. It is difficult for students to have a sense of what it is they are protecting and who or what they are protecting it from.

Security choices are often a risk management tradeoff between threats and the costs of deploying protection mechanisms, including detrimental effects on productivity resulting from security choices. Unless the student has a sense of the value of what is protected, and an understanding of the strength of motive of potential attackers, it is hard to gauge the required strength and assurance of protection mechanisms. Similarly, it is hard to appreciate the effects of

security choices on costs and productivity unless there is a constrained budget and users who must directly or indirectly interact with the protection mechanisms to productively utilize the computing resources. CyberCIEGE is designed to immerse the player in this risk management context. Player choices have implications for both the protection of information, and costs to a virtual enterprise.

2. A LANGUAGE FOR EXPRESSING A RANGE OF SCENARIOS

Early in the project we concluded that a range of different risk management contexts are needed to illustrate how the effects of security choices depend on the security policy and the physical environment. To achieve this, we defined a language with which to express the security related risk management tradeoffs for different scenario contexts, which we call "scenarios". The CyberCIEGE game engine interprets this scenario definition language and presents the player with the resulting game. What the player experiences in a given instance of the game and the consequences of the player choices are a function of the scenario as expressed using the scenario definition language.

2.1 Security Policy

The question of whether a system is secure depends in large part on the security policy. Stated another way, a system can only be said to be secure with respect to some well-formed policy expressed in terms of information and authorized user access to information. The CyberCIEGE scenario definition language captures the abstraction of information security policy through a construct called an "Asset", which represents information of some value to the enterprise. The scenario designer defines multiple assets, each having a set of attributes (e.g., the value of the asset, who is authorized to access the asset, etc.). Taken together, these asset definitions reflect the abstract information security policy for the enterprise modeled within the scenario.

Expressing an abstract information security policy requires identification of what is being protected from whom. When an asset is defined in the language the core notion of "what" is expressed as a cost to the enterprise in the event of some form of compromise. For example, "the enterprise would loose the equivalent of $500,000 should the secrecy of this asset be compromised." Or, "the enterprise would loose the equivalent of $100,000 should the integrity of this asset be compromised". Similarly, the notion of "whom" is expressed as a motive for some form of compromise. For example, "an attacker has a very high motive to succeed in compromising the secrecy of this asset.

The language represents loss to the enterprise in terms of dollars, which is the metric by which the player's success is measured. The scenario designer controls the initial amount of money available to the player, and a monthly budget that fluctuates based on the productivity of the virtual users within the game. The scenario designer also controls the monetary loss resulting from compromise of assets. Thus, the scenario designer determines the impact of any given compromise on the success or failure of the player. Some loses can result in annoyance. Others bankrupt the entire enterprise.

Motive is expressed as a value ranging from zero to one thousand. Motive represents the lengths to which an attacker will go to compromise the asset. Table 1 summarizes motive values as they are used in the game. The higher the motive, the greater the protections needed to prevent compromise of assets. Motives of 800 and higher result in professional attacks utilizing subversion [2].

Table 1. CyberCIEGE Attacker Motives

Motive Value	Description
1000	Unstoppable, you trust no one.
800	Major interest to professional attacker
400	Strong interest to professional attacker
100	Minor interest to professional attacker, major interest to skilled hacker.
50	Minor interest to skilled hacker, major interest to unskilled hacker
25	Minor interest to unskilled hacker
0	Nobody will even read your blog.

2.2 Expressing Different Kinds of Policies

There may be different motives for compromising the same asset, with differing consequences. For example detailed test data for a secret formula might be of great value to a competitor, with catastrophic consequences to the enterprise from its compromise. On the other hand, a rival engineering manager within the same enterprise may be motivated to gain access to the same test data for purposes of showing how far behind the project is, with the intent of taking it over. To reflect these differing motives and costs, each asset has an associated "cost list" reflecting potentially different costs and motives for different "attackers". Cost lists also can distinguish between different modes of compromise, permitting the scenario designer to differentiate between damage resulting from disclosure and damage resulting from unauthorized modification.

Additionally, each asset may also have a secrecy label and/or integrity label to reflect labeled information security policies. These policies are sometimes called "mandatory access control (MAC) policies"[3]. The asset label attributes name "secrecy" and "integrity" constructs that are also part of the scenario definition language. For example, the scenario designer can define a secrecy level of "Secret, and give it attributes reflecting a loss to the enterprise resulting from disclosure and a motive for attackers to disclose any asset with that label.

Within scenarios that include labeled assets, the costs and motives associated with the "enemy" (e.g., a ruthless competitor) are reflected in the secrecy and integrity labels. And the costs and motives associated with internal intrigues, rivalries, ineptness and informal "need-to-know" are reflected in the explicit cost lists associated with each asset. In traditional computer security jargon, the cost lists are intended to reflect discretionary access control (DAC) policies while the labels reflect mandatory access control policies.

In addition to cost-list and label-based costs and motives, each asset includes an attribute reflecting the damage to the enterprise should the asset become unavailable due to a denial of service attack. Assets also have a corresponding denial of service motive.

Potential attackers (i.e., those motivated to compromise assets) include unauthorized enterprise users named in asset cost lists, and "external" attackers. Within the game, the means employed by attackers to compromise assets is a function of their motive. And, with suitable motive, external attackers will bribe internal enterprise users (i.e., the veritable "insiders") to compromise an asset. This includes users who are authorized to access an asset as well as unauthorized users who may have to defeat protection mechanisms to compromise the asset. Thus, CyberCIEGE represents "insiders" in two ways: those named in asset cost lists with explicit motives to compromise assets for which they are not authorized, and those who are bribed or otherwise coerced by external attackers.

2.3 User Trustworthiness and Insider Attacks

No amount of technology-based protection mechanisms can defend against an insider who can be bribed into compromising assets for which the insider is authorized. The virtual users within CyberCIEGE have a scenario-defined "trustworthiness" and a player-selectable "background check". The player can purchase different degrees of background checks for different groupings of virtual users, thereby reducing the vulnerability of users to bribes. For example, those users who are authorized to view valuable secrets can be given high background checks while other users have low or no background checks. As a resource, background checks are relatively expensive, so the player is dissuaded from purchasing background checks for all users who might gain access to the sensitive assets.

The modeling of background checks within CyberCIEGE is not entirely consistent with real world behavior. This results from a few bounds placed on the game development intended to limit complexity of game play. For example, the virtual users are static in each scenario. The player does not hire or fire users (though the player does hire and fire IT support staff and guards). Also, the player does not assign work to useres. The scenario designer defines which assets each user must access to perform work. Thus, if a user is not trustworthy, the player can't fire the user or assign the user to low risk assets. However, the player can purchase background checks. Doing so magically improves the user trustworthiness – but at a monetary cost. This level of modeling is sufficient to illustrate the value and the cost of background checks.

3. ENFORCING THE POLICY WHILE GETTING WORK DONE

To succeed in the game, the player assesses the information security policy of a given CyberCIEGE scenario by understanding:

1) The trustworthiness and asset authorizations of the users; and,
2) The costs to the enterprise resulting from asset compromise and the motive of attackers to achieve compromise.

. The other half of the risk management tradeoff is the need for users to access different assets to do their jobs. CyberCIEGE includes a construct called an "asset goal", which the scenario designer uses to reflect productivity losses resulting from an inability of users to access assets needed to be productive. Within the game, a given asset is in one of two places: either in a user's head or on a computer component (e.g., workstation or server). Users can only achieve asset goals if the assets are on computer components. Thus, the player must provide the users with computer components with which to process the assets.

3.1 Productivity Requires Assets on Computers

The game includes a catalogue of computer components that player can purchase to satisfy the virtual user needs to process assets. Some scenarios start with components that initially contain assets. Other scenarios might start with no components, requiring the player to purchase and configure components. Figure 1 illustrates a CyberCIEGE virtual user who has been provided a workstation with which to access assets.

Figure 1. CyberCIEGE User Productively at Work

Many users may need to access the same asset to achieve their respective goals. This drives the player to connect computer components together using networks. Players can select from multiple different networks, including the Internet. The player can purchase network devices such as routers and firewalls to interconnect networks.

Each asset is instantiated on at most one component. Left to their own devices, users will store assets on components that are most efficient for all the users to access. The player must make choices to constrain which assets are processed on which computers.

Each user can have multiple asset goals. Each goal identifies one or more assets that must be accessed by the user to achieve the goal. When multiple assets are named in the same goal, the user must be able to concurrently access the assets from the same workstation. This allows the scenario designer to require the player to achieve a form of controlled sharing of assets across a range of security labels, resulting in a need for processing in a multilevel mode [3].

Optionally, asset goals also identify software applications that must be used when achieving the goal (e.g., require the use of an email client to access email assets). CyberCIEGE includes a catalogue of software applications, some of which are only available on certain operating systems. In some cases, there are multiple products of the same type of application, e.g., different email clients. And some products are vandalized more frequently than others. Also, to support information integrity scenarios, some applications and operating systems have more integrity than others. This can be used to illustrate the risks of low integrity software accessing high integrity assets [4].

Scenarios can be constructed such that failure to achieve one asset goal may be very costly to the enterprise, while failure to achieve a different goal results only in an unhappy user. Each asset goal includes a productivity attribute and a happiness attribute. The scenario designer can introduce goals that encourage the player to make unwise security choices in response to user

complaints. In such a scenario, the player may have to live with unhappy users to maintain security.

Asset goals can be dynamically introduced into a scenario based on current game conditions, e.g., the passing of time or a degree of success by the player. Altering asset goals is the primary means by which the scenario designer engages the player in an ongoing narrative. These changed goals can lead the player to buy new components or software, or make new network connections such as risky connections to the Internet.

3.2 Protecting Assets on Computers

Components have different security properties, e.g., operating systems that enforce policies with differing levels of assurance. Also, the player can choose to alter the security configuration of components. For example, workstations can be configured to automatically log off users after a period of inactivity. And players can set procedural security policies such as prohibitions against writing down passwords and the frequency of anti-virus updates. These of course depend to a great extent on user training, which is another set of choices the player can make. The variety of choices a player can make that potentially affect the security of components is shown in the screen capture of figure 2.

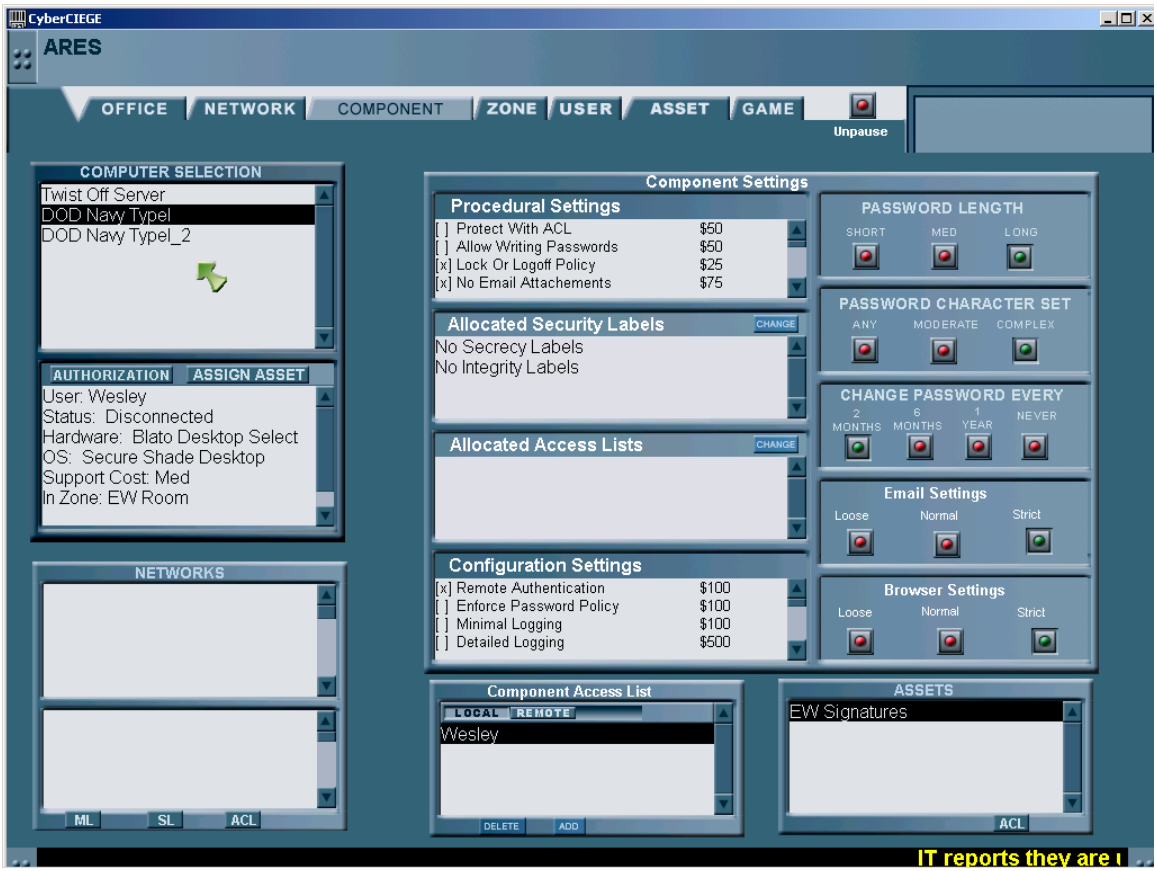

Figure 2. CyberCIEGE Component Configuration Settings

The operating systems on CyberCIEGE components affect the security of the component and constrain the kinds of applications available for use on the component (e.g., may limit what

applications can run on a specific server). Most operating systems can be configured to enforce a DAC policy. Some can be configured to enforce a MAC policy. And some operating systems have more assurance than others. The MAC operating systems include multilevel network connections and single level network connections. The player is responsible for selecting the security attributes of network connections to components.

Components include local and remote user accounts, allowing the player to constrain which users can access which components. Of course the enforcement of these constraints depends on factors such as operating system assurance and IT support staff.

3.3 Physical Protection of Assets

The extent to which computer component protection mechanisms are relied upon to enforce the security policy depends in large part on the degree of physical security. In some environments it is possible to store assets on components having very weak security, e.g., if the component is in a locked vault with no network connections. Each game scenario includes one or more physical zones that can be used to control the physical movement of users. An example of a zone is a physically secure office with a locked door for which only selected users have a key. When components are purchased, they are placed within a specific zone. Physical access to components can therefore be constrained based on the physical access to the zone. Players can increase the physical security of a zone by purchasing items such as guards, cipher locks and alarms. Players also choose who is permitted to enter the zone, and place constraints on what a user can carry into and out of the zone (e.g., cell phones and cameras).

Networks can extend between zones, resulting in an opportunity for attackers to engage in wiretap attacks. CyberCIEGE components include VPN gateways and link encryption devices that players can deploy to protect against such attacks.

4. CONCLUSION

The CyberCIEGE scenario definition language can be used to express an information security policy and define a work environment where user access to assets is necessary to maintain productivity. This allows a scenario designer to force the player to address the fundamental tension of computer security: Provide users with suitable resources to productively perform work, while protecting the assets from security compromises in accordance with the enterprise security policy.

REFERENCES

[1] Irvine, C. E., and Thompson, M., Teaching Objectives of a Simulation Game for Computer Security, *Proceedings of Informing Science and Information Technology Joint Conference*, Pori, Finland, June 2003.

[2] Anderson, E. (2002, March). A Demonstration of the Subversion Threat: Facing a Critical Responsibility in the Defense of Cyberspace. Masters Thesis. Monterey, CA: Naval Postgraduate School.

[3] Brinkley, D.L. and Schell, R.R. (1995). Concepts and Terminology for Computer Security. Information Security. ed. Abrams, Jajodia, and Podell. Los Alamitos: IEEE Computer Society Press. Retrieved November 21, 2002 from World Wide Web http://www.acsac.org/secshelf/book001/02.pdf

[4] Irvine, C., and Levin, T., "A Cautionary Note Regarding the Data Integrity Capacity of Certain Secure Systems," in Proceedings of the Working Conference on Integrity and Internal Control, Brussels, Belgium, 15 November 2001.

HONEYNET SOLUTIONS

A deployment guide

Ronald C Dodge JR, Richard T Brown, Daniel J Ragsdale

United States Military Academy

Abstract: Honeynets provide network and system managers a unique intrusion detection and monitoring system that provides indications of malicious behavior in a near "false positive" proof manner. When deployed properly, these systems can provide warning of both inside and external network threats. However if the deployment is not tightly integrated into the existing topology and the honeynet is configured to allow in only the threat intended to monitor, the usefulness will be greatly diminished. We propose a number of honeynet deployment schemes to answer some specific questions about their uses, capabilities, and resource requirements. Specifically, the paper addresses deployment of a Gen II honeynet, a solution for deploying a honeynet to a remote location, a consolidated honeynet scheme, and a model for targeted honeynets.

Key words: Honeynet, Honeyfarm, Intrusion Detection

1. INTRODUCTION

When it comes to internet-based attacks, computer network system administrators are concerned with three main principles: detection, prevention, and response. A system administrator who believes there is malicious activity on one of his networks will often consult a server, firewall, or host computer event log. Unfortunately, separating malicious connections and events from legitimate activity can be a next to impossible task – like finding a needle in a haystack. Honeynets are an extremely useful security tool because they allow system administrators to go from finding a needle in a haystack to finding a *needle in a needle stack*.

Honeynets focus on the principles of detection, response, and monitor. They are designed to be probed and attacked. All data collected is of high value because honeynets have no legitimate production traffic. The data collected is "unpolluted" by other activity. Thus, we are searching for needles in a needle stack. Honeynets are an ideally suited security tool for detecting a new internet based attack and catching advanced attackers. They will reveal the identity of the attacker, the attacker's means of communication, and the tools used in the attack. Depending on the type of attack, the signature can then be added to firewall and intrusion detection system databases or operating system software can be patched to remove a vulnerability. Ultimately, honeynets contribute to the prevention of future attacks.

Honeynets can be categorized into low and high interaction categories; we will address high interaction honeynets. In a high interaction honeynet, the server and host systems have real

operating systems and applications. These systems require extensive resources and maintenance, to include monitoring – remember, you are pacing a vulnerable system(s) on a segment of your network with the hopes it will be probed and attacked. The time required to monitor honeynets is often what makes system administrators resist using them. One of our proposed solutions includes the incorporation of a centralized honeynet system (honey farm) for large organizations to relieve the honeynet monitoring burden on the local system administrator.

The deployment of a honeynet is a risky endeavor because it has the capabilities of a full production network. Extensive risk mitigation measures must be performed to prevent a honeynet computer from being used in an attack inside your network or on outside systems. An example would be a firewall system (or as described later, a honeywall) set to drop or modify outbound packets that match a known malicious signature. Risk mitigation is essential for data control and will be discussed in this paper.

We present four deployment schemes for honeynets that solve some very specific problems. The deployment schemes are a Generation II honeynet, a solution for deploying a honeynet to a remote location, a consolidated honeynet scheme, and a model for targeted honeynets.

2. CHARACTERISTICS OF A GEN II HONEYNET

Until recently, honeynet sensors would perform bridging at layer 3 of the OSI model. Conceptually, this makes sense because the IP header contains the information necessary for routing to a honeypot system at a certain IP address. Data control was performed by a standard firewall operating at layer 3 as well. These systems operating at layer 3 are referred to as Generation I honeynets [1]. While this method of deployment works, an advanced attacker could discover in very short order that he was not on a real production system or the system was being used to monitor his actions. The potential consequences of this discovery could prove disastrous to the organization launching the honeynet. For example, a vengeful attacker could launch a distributed denial of service attack on that organization.

An important question arose from the previous scenario. Can a honeynet be deployed that is less visible to an attacker and still performs the critical tasks of data capture and data control? Gen II honeynets are the answer to this question. Gen II honeynets have a single sensor that bridges at layer 2, as shown in figure 1. This gives two main improvements over Gen I honeynets. First, the IP header is not disturbed because Gen II honeynets operate below it. Essentially, they are routing traffic at the MAC level. This routing is invisible to an attacker because there are no TTL decrements in the IP header.

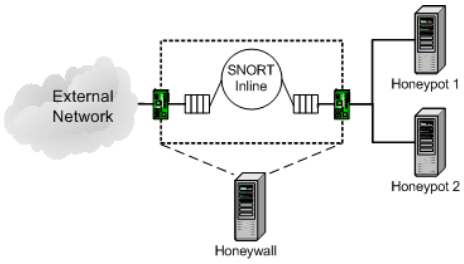

Figure 1: Honeywall Schematic

Second, a Gen II honeynets can be part of a production network. The layer 2 operations allow us to covertly insert a single honeypot into a real production network with minimal risk to the other systems in that LAN. This makes the existence of a honeypot less distinguishable to an attacker.

A honeywall performs data control for Gen II honeynets. Data control for a honeynet should allow traffic in, but carefully restrict the outbound traffic. For example, a honeywall could be configured to only allow a max of 10 outbound connections. This would protect against the machine being used for denial of service attacks. Additionally, the honeywall runs a modified version of SNORT called SNORT-Inline that utilizes the queuing feature in IPtables/EBtables to inspect every outbound packet. If the packet matches a known exploit (or any other rule the honeynet manager want to implement) one of four actions can be taken. First the packet can be allowed to pass on to the external NIC; second the packet can be dropped, third the packet can be rejected; or fourth (and more interestingly) the packet can be modified to render the exploit useless.

While the honey wall can capture all packet data in a tcpdump format and log alerts based on a snort rule set, the most valuable data is captured on the individual honeypots. If an attacker is operating in an unencrypted mode, his data can be captured using any one of several packet utilities to reassemble TCP packets. However, many attackers today encrypt their attack communication. This presents a significant challenge to data capture. One method would be to record the encrypted packets and attempt to break the encryption scheme. This can be very tedious and is not preferred. The Honeynet Project (www.honeynet.org) has developed a tool called Sebek for data capture of encrypted channels. [2]

Sebek operates on the principle that it is better to circumvent session encryption with a kernel based method rather than break the encryption. Sebek software operates at the kernel of an operating system, making it virtually invisible to an attacker. It circumvents encryption by recording an attacker's keystrokes and recording the packet stream as it is being decrypted and reassembled. Some of the information that Sebek is designed to capture or recover are files copied with SCP, passwords used to log in to remote system, passwords used to enable Burneye protected binaries, and other forensics related tasks.

The combination of using a honeywall and sebek provides a sound method of security in the honeynet system through data control and data capture. The data captured is logged and stored on a separate, secured system to prevent loss of data in case attacker is alerted to honeypot.

3. HONEYNET DEPLOYMENT SCHEMES

Honeynets can be deployed in many different ways. Each different implementation should be examined and used to meet specific security needs. In this section we present different ways that can support both economic and mission requirements.

3.1 Consolidated Honeynets (Honey Farms)

System administrators often have too little time and resources to devote to managing a honeynet. An organization with extremely limited time and resources for information security should focus on intrusion detection systems, firewalls, and sound security practices first. In general, honeypots should only be used in conjunction with other security mechanisms. Although the local system administrator may not have the time to dedicate to a honeynet, the benefits of honeynet deployment are far too great to let fall to the chopping block. A distributed

organization can consolidate its honeynet traffic in one physical location that is monitored by dedicated professionals 24/7. This concept is called a honey farm, shown in Figure 2.

Figure 2: Model of a consolidated honeynet with a honeyfarm

Honey farms consolidate all honeynet traffic by becoming the destination for scanned IP addresses that are owned by an organization but not in use in a real production system. All honeynet traffic is redirected from a router outside of a production system to a router outside of the honey farm though a Generic Routing Encapsulation (GRE) tunnel. Cisco developed this protocol to allow an arbitrary network protocol A to be transmitted over any other arbitrary network protocol B, by encapsulating the packets of A within GRE packets, which in turn are contained within packets of B [3]. A GRE tunnel achieves many of the same advantages of the routing done by a Gen II honeynet. Both protocols operate at layer 2. A GRE tunnel, through its encapsulation, encloses the IP header of the original packet and tricks the TCP/IP protocol into only performing one TTL decrement of the payload packet. When applied in the honey farm scenario, there is no decrement of the TTL of payload packet until it arrives in the honey farm where the actual IP address being attacked is physically located.

In addition to data control and data capture, honey farms perform the additional role of data collection. All the honeypot computers for a distributed organization are physically located together making collection and analysis of the data an easy endeavor.

A very robust application of the honey farm for an enterprise involves dynamically reconfiguring each router to account for differing threat levels across an enterprise. An enterprise located in three different countries may experience a sudden increase in the number of scans and attempted attacks from one of the countries. By dynamically configuring the routers, an enterprise can have its routers assign more of the honeynet IP addresses to the country under attack. This will enhance the attack data received and give a greater probability of capturing a new attack.

3.2 Deployable Solution to a Remote Area

In today's global mobility, an organization will often set up a new base of operations in a foreign country. It is necessary to network operations in that new "remote" location to the headquarters and this is usually accomplished over the Internet. While global expansion is usually good for business, there are some risks in the modern world of terrorism and hackers. Any organization (commercial business, government, military, humanitarian, etc.) must ask itself an important question – "Am I being specifically targeted based on my deployment or expansion into a new geographic area?" A deployable honeynet can be a great tool to help answer this question.

Initially, an organization will set up a LAN with a small footprint at its remote location. The remote site can incorporate a small honeynet or virtual honeynet using a laptop with technology such as UML, HoneyD, or virtual machines. If the honeynets at the remote site and back at the headquarters capture the same types of attacks from the same SRC, there is good reason to believe that the organization is being specifically targeted. Specific targeted computer attacks are usually performed by an advanced attacker. It is for this reason that a high interaction honeynet vice a low interaction honeypot should be incorporated at both sites.

The organization may want to incorporate the consolidated honeynet technology from the previous section. The system administrator at the remote site will probably not have the time to constantly monitor and conduct analysis of the data captured from the honeynet. A GRE tunnel can be implemented from the remote honeynet back to a honey farm at the headquarters (see Figure 2).

3.3 Targeted Honeynets

The deployment of honeynets involves more than a network topology template. The identification of the type of malicious activity that is desired to be observed is integral to the effectiveness of a honeynet. The concept of a honeynet is that all data observed on the honeynet is suspicious. So the normal activity of examining IDS logs looking for malicious activity within normal traffic is likened to looking for a needle in a haystack; a honeynet turns this into looking for a needle in a needle stack. The idea of a "tuned honeynet" is to reduce the number of needles. As an example; the security level, topology, and content is significantly different if your target is a "script kiddie" than if it is a sophisticated intruder. This can be predicated on specific intelligence on the potential intruder, although it is not required. If a certain subnet of an organizations network has been compromised and an expectation exists that the attacker might continue to other network segments, then a honeynet can be established that presents a very similar configuration as the one the attacker already compromised. If no confirmed intrusions have been documented, a honeynet can be established to present to an attacker the activity that is most critical to your organization. For example, for an organization that has valuable proprietary data, the honeynet would be constructed to

• Be fairly secured. Lesser skilled attackers would not be interested in the organizations data – they are more interested in credit card accounts, hard drive space, and installing relay bots. The security level should be high enough to "screen out" these attackers. Attackers with a focus on state or corporate sponsored espionage should be able to bypass most normal security techniques.

• Be a believable topology. The topology should not present just a single machine on a subnet. Populate the honeynet with services and systems that mimic a real deployment. As an example, the topology might place a file server on the same subnet of a vulnerable web server. This subnet could be placed behind a firewall with an open port that an attacker could use.

• The content of the web and file server should be believable. A honeynet without honey will provide only limited information about the attacker.

Sometimes we want to create a honeynet that is designed to catch a certain kind of attacker. The level of security required (S, S', S'', etc) depends on the threat and the data being "protected." Some examples are:

- S -- Script Kiddie – P2P Server (Kazaa, etc)
- S' -- Credit Card DB
- S' -- Military Troop Info
- S'' -- Chem-Bio Warfare Info

4. CONCLUSION

Honeynet use is rapidly increasing. In May 2004 the Honeynet Project released a honeywall CD Rom that was downloaded over 150,000 times in a two week period. Honeynets are capable of capturing attack data that may otherwise go unnoticed. The deployment of honeynets should be focused on the threat that an organization is interested in tracking. As honeynet use continues to grow, attackers will surely develop ways to detect their use. Operating at layer 2 provides a method of hiding the presence of the honeynet sensor, but this will only be temporary until attackers find a way to detect layer 2 routing. Security professionals everywhere should constantly strive to invent new innovative ways to outperform the attacker community. Ways to mitigate the resources required to implement honeynet solutions include honeyfarms and deployable "honeynet in a box" solutions that need only be connected to a network.

5. REFERENCES

[1] Know Your Enemy: Honeynets, 12 November 2003. Retrieved off the world wide web from the Honeynet Project www.honeynet.org/.

[2] Know Your Enemy: Sebek, 17 November 2003. Retrieved off the world wide web from the Honeynet Project www.honeynet.org/.

[3] Spitzner, Lance. Honeypots: Tracking Hackers, Pearson Education, Inc., 2003.

THE ADVANCED COURSE IN ENGINEERING ON CYBER SECURITY

A Learning Community for Developing Cyber-Security Leaders

Kamal Jabbour and Susan Older

Syracuse University and Air Force Research Laboratory

Abstract: The Advanced Course in Engineering on Cyber Security (ACE-CS) is a public-private partnership to develop top ROTC cadets into the next generation of cyber security leaders. Modeled after the General Electric Advanced Course in Engineering, ACE-CS immerses students in the cyber-security discipline through a combination of intense coursework, open-ended problems, and concurrent internships. In this paper, we discuss the ACE-CS pedagogy, the successes and challenges of its inaugural offering, and some future directions for the program.

Key words: Cyber-security education, technical leadership, learning community.

1. INTRODUCTION

The objective of the Advanced Course in Engineering on Cyber Security (ACE-CS) [1] is to develop the next generation of cyber-security leaders, with a particular emphasis on educating future military leaders. Through a public-private partnership among the Air Force Research Laboratory (AFRL), the US Military Academy, and Syracuse University, ACE-CS follows the model of the General Electric Advanced Course in Engineering [2] to help transform top cadets in the Reserve Officers Training Corps into original thinkers, problem solvers, and technical leaders.

The underlying philosophy of ACE-CS is to completely immerse students in the cyber-security discipline, through a combination of intense coursework and internship experiences. Each week, students attend a daylong lecture, given by a domain expert from the military, academia, or industry. They also spend three days a week in cyber-security internships, at either government labs or local industry. In addition, they work in teams to solve open-ended, real-world problems; they then write individual reports to present their solutions.

This paper presents the underlying pedagogy of ACE-CS, discusses the successes and challenges of its inaugural offering, and outlines some future directions for the program. Specifically, Section 2 describes the program's educational objectives and its approach to meeting those objectives. Section 3 provides more details about the content of the course, including sample real-world problems assigned to students. Section 4 details some of the results of the initial (2003) offering of ACE-CS, as well as adjustments that will be made in the 2004 offering. Finally, we conclude in Section 5 with a summary of the factors that we believe have most influenced the success of ACE-CS.

2. EDUCATIONAL OBJECTIVES AND APPROACH

Critical to the success of any academic program is to identify the desired educational outcomes. Focusing on our expectations for students guides us in developing appropriate learning experiences, selecting topics for inclusion, and assessing student success [3].

The goal of ACE-CS is to develop original thinkers and technical leaders who can solve real-world problems in the area of cyber security. Specifically, when faced with a real-world problem, ACE-CS graduates must be able to do all of the following:

1. Formulate a clear problem statement.
2. Make reasonable assumptions about the problem context.
3. Apply sound analytical techniques and engineering tools.
4. Solve the problem to a specified depth.
5. Perform risk analysis on the solution.
6. Deliver a solution on time.
7. Communicate that solution effectively, both in writing and orally.

2.1 Program Approach

The ACE-CS program structure directly reflects its educational objectives. Modeled after the 80-year old General Electric Advanced Course in Engineering (now known as the Edison course [2]), the program combines (1) an intense classroom environment with real-world problems, (2) mentoring by experienced cyber security professionals, and (3) real-world experience through internships. The overall program–viewed separately from the specific course content–forms a learning community [4] centered on cyber security.

The course itself meets for eight hours once a week. A typical class begins with the timely submission of written reports and the oral presentation of solutions for the previous week's problem. Cadets discuss their solutions with the ACE Director and the instructor, before moving on to a new topic. Each week brings a different instructor, who assigns a substantial real-world problem for the next week and lectures for six hours on the background material for that topic. The instructors–drawn from government, academia and industry–are chosen for their expertise in the given topic.

Cadets work on teams of three to solve the assigned problems, which typically require 40-80 hours per team to solve. They then write and submit individual reports. In addition, each team must give two structured presentations during the ten-week course, one using slides (e.g., PowerPoint) and one using chalk on a blackboard. The presentations provide cadets experience in articulating, justifying, and defending a particular technical point of view. The presentations are nominally 15 minutes in length. However, they typically spark open debates among the class, as different teams attack the validity of others' assumptions and solutions.

Three days a week, cadets work with mentors at local private or government cyber-security laboratories during the day. These internships expose the cadets to the practical challenges of cyber security and help them establish professional relationships with domain experts. The ACE-CS Director matches students and internship opportunities before cadets arrive in June, based on employer needs and student background. Employers provide a paragraph describing the tasks they have for an intern, and students provide a 100-word bio describing their background and interests. Companies such as Par Technologies and Dolphin are looking for civilian students who may be interested in working for them full-time after graduation. Classified labs such as the Northeast Air Defense Sector (NEADS) need students who already have security clearances.

Fridays generally provide the military component of the ACE-CS experience. In addition to a weekly 8-mile run with the ACE-CS Director, cadets participate in flag ceremonies on base. There are also several field trips to military installations. For example, the 2003 cadets visited Fort Drum to observe the Phoenix Warrior live war games, the 174th Air Wing of the National

Guard in Syracuse to see an operational Air Wing, and NEADS to observe the operation of a net-centric command and control center.

2.2 Student Assessment

The written reports serve as the primary assessment mechanism for gauging student progress. Although there is no mandated length for the reports, they typically run 30-40 double-spaced pages. The ACE-CS director and the instructors evaluate the reports with respect to the desired educational outcomes. Specifically, each report is graded on a 100-point scale, with the following weights: 10 points for the problem statement, 10 points for quality assumptions, 10 points for the use of analytical techniques and tools, 20 points for the solution itself, 10 points for the risk analysis (i.e., determining how dependent on the initial assumptions the solution is), and 40 points for the quality of writing (e.g., style, grammar, neatness, format, references). Students receive zero credit for a report not submitted on time; a second late submission results in expulsion from the program.

Like the reports, presentations are evaluated for both their content and their adherence to a strict format. PowerPoint presentations are limited to seven slides, and the first three slides must provide (respectively) a clear statement of the problem, the assumptions upon which the solution depends, and a summary of the tools and techniques employed in solving the problem. The remaining slides are devoted to the solution itself.

Cadets also evaluate themselves and their peers at the end of the course. Specifically, they must indicate what each team member's contributions were, and what percentage of the work each member performed. These evaluations influence the final grade that students receive for the course, which carries four credit hours of academic credit from Syracuse University. Students who successfully complete the program can apply the earned credit towards their programs of study at their home institution.

3. COURSE CONTENT

ACE-CS was first offered in Summer 2003, with an enrollment of seventeen students from across the country: twelve Air Force ROTC cadets, two civilian undergraduates, and three civilian graduate students. (In this paper, we shall follow the ACE-CS lead and use "cadets" to refer to all students, regardless of their ROTC status.) All but one cadet had completed at least three years in a computer-related discipline (electrical engineering, computer engineering, computer science, or information studies) and had experience in both programming and operating systems. Previous networking experience was desirable, but not necessary. The average grade-point average (GPA) was 3.2, on a 4.0 scale.

There were ten separate lectures, each covering a different aspect of cyber security. The lectures primarily focused on technical aspects, but they also covered legal and policy aspects of security. The full assortment of topics, along with the instructors who taught them, appears in Table 1. In addition to lecturing, the instructors also designed the problems that cadets worked on for the next week. These open-ended problems reflected the sorts of situations that cyber-security professionals encounter in the real world.

Table 1. Week-by-week syllabus of the ACE-CS course.

Week and topic	Content	Instructor
1. Legal Issues	Internet laws and cyber crime, the Fourth Amendment of the US Constitution, search and seizure of data, rights and privacy issues, government versus private workplace, search warrants and wiretap laws, the PATRIOT Act.	Prof. Lisa Dolak, *SU Law Professor*
2. Security Policies	Establishing and implementing security policies, confidentiality integrity and availability considerations, identifying	Joseph Giordano, *Technical Advisor,*

Week and topic	Content	Instructor
	vulnerabilities and threats, establishing disaster response and recovery procedures.	*Information Warfare Branch, AFRL* LT Chad Korosec, *US Naval Reserve Officer*
3. Cryptography	Mathematical basis for data encryption, substitution ciphers and the Data Encryption Standard, private-key and public-key cryptography, key distribution and trusted authority, digital signatures.	Prof. Shiu-Kai Chin, *SU Professor*
4. Computer Security	Operating systems and file system security, passwords and one-way hashes, user-space administration, archiving and back-up strategy, intrusion detection, disaster response and recovery.	Prof. Steve Chapin, *SU Professor*
5. Digital Forensics	Procuring and analyzing digital evidence, preserving the chain of custody of digital evidence, recovering hidden data on hard drives, classifying file systems, analyzing slack and sector data, recovering lost clusters.	Mr. Chet Hosmer, *CEO of Wetstone Technologies*
6. Network Security	TCP-IP packet format and vulnerabilities, protocol and implementation flaws, buffer overflow, denial-of-service attacks, distributed attacks, email, domain name system, web servers.	Prof. Heather Dussault, *SUNY-IT Professor*
7. Steganography	Data hiding in images, classifying steganography algorithms and tools, categorizing vessel capacity, detection and recovery of hidden data, digital watermarking, streaming media steganography, multilingual steganography.	Dr. Leonard Popyack, *Director of Adversarial Science Unit, AFRL*
8. Network Defense	Host and network security, firewalls and periphery intrusion detection systems, bastion hosts, network monitors and traffic analyzers, network logfiles, detecting anomalous behavior, network recovery.	Lt. Col. Daniel Ragsdale and Major Ronald Dodge, *United States Military Academy*
9. Wireless Security	Wireless local area networks, wireless encryption protocols, wardriving.	Mr. Paul Ratazzi, *AFRL/IFGB*
10. Next-generation Cyber Security	Next-Generation Internet Protocols IPv6, embedded systems, 3G cell phones and personal data assistants.	Prof. Kamal Jabbour, *SU Professor*

For example, the *Security Policies* problem required cadets to develop the security policies and procedures for an Air Force Air Operations Center (AOC) that contains a weather cell, a logistics cell, a Command and Control (C2) center, and an intelligence-gathering and processing center. The different cells and centers must operate at different security levels. In addition, the C2 center involves primarily Air Force personnel, but also includes Army, Navy, and some British military personnel. Given an initial high-level AOC architecture, cadets had to review and appropriately modify the architecture; perform a risk assessment on the AOC; develop a security architecture to overlay on the AOC architecture; and develop and define the AOC's security policies and procedures.

For the *Computer Forensics* problem, the instructor completed his lecture by tossing a USB thumb drive on the table. He informed the cadets that customs officers had seized it from a suspected drug dealer trying to enter the United States at Niagara Falls. He then asked the cadets to analyze the drive for (simulated) evidence that might support an indictment. To accomplish this task, the cadets faced the challenge of making 5 copies without modifying the device, calculating a hash, making assumptions, analyzing files and images, recovering stegoed data from an image of the Falls, restoring deleted email from the drive slack, translating foreign information, interpreting addresses and identifying the country, mapping drug slang into English, and then compiling a long list of circumstantial and forensic evidence.

The background scenario for the *Wireless Security* problem made the cadets part of an Air Force Office of Special Investigation (AFOSI) cyber-crime team attempting to gather preliminary evidence in a computer-crime investigation. Cadets were given authorization to search the Air Force premises of Griffiss Business & Technology Park in Rome, NY to find a hidden wireless

network. To ensure that they did not accidentally start analyzing the wrong networks, cadets had to report the location and identifying information of the suspected network to Air Force authorities. Once given the authorization to continue, cadets could move on to their primary task, which was to use any available tools and techniques to gather as much information as possible, including network configuration, identity of devices on the network, and contents of the data on computers and traversing the network. For their reports, students had to document their procedures and findings, and also determine whether the various vulnerabilities they found were fixable by the (hypothetical) criminal suspects or represented inherent vulnerabilities of the technology.

4. EXPERIENCES AND ADJUSTMENTS

All seventeen students successfully completed the 2003 offering of the course, and all but one student received an A or B. Eight of the nine problems were solved by all teams. However, only one team completely solved the *Wireless Security* problem, which ultimately required the cadets to find the hidden network (i.e., wardriving), determine the level of encryption, capture enough packets to crack the encryption (around 4 million packets, which took several hours), outline the topology of the network, find vulnerabilities (a misconfigured FTP server), compromise and penetrate the server, and then recover a password from a hidden directory. Only one team realized that they needed to write a C-program to strip the unencrypted headers from the encrypted packets before attempting to crack the encryption.

During the summer, cadets requested to play a cyber war game. On their own initiative, and with logistical support from the engineers in the Next-Generation Cyber Security Laboratory at AFRL, the cadets divided into a Blue Team and a Red Team. The Blue Team designed the target network, with a strategically placed series of "flags" (i.e., vulnerabilities). The Red Team deployed an extensive arsenal of off-the-shelf and custom computer-network attack tools, and systematically attacked the blue network over the course of one Friday. John Gilligan, Chief Information Officer of the US Air Force, met with the cadets near the end of the exercise and discussed their ACE experiences.

The internship component was very successful. Two cadets interned at NEADS, two cadets interned at local companies, and the remaining cadets worked in various labs throughout AFRL. The only complaint expressed by employers was a desire to have more of the cadets' time allocated to the internships.

The cadets were very candid in their ACE-CS evaluations. They expressed strong views about various instructors, particularly complaining about those problems they found insufficiently challenging. Throughout the summer, they also expressed unhappiness with the ACE-CS Director's insistence on correct written English and the stringent writing criteria he imposed. However, this reticence slowly turned into self-congratulatory praise as they appreciated their newfound ability to communicate professionally. The cadets also expressed appreciation for the 8-mile runs and indicated that the runs should remain a component of the program.

The 2004 offering of ACE-CS begins in early June. Out of 57 applicants, a total of 26 students will be participating in the program. Of these, seventeen are ROTC cadets (14 Air Force, 1 Navy, and 2 Army), and nine are civilian students. The average GPA is 3.4. All but two of the students are in computer science, computer engineering, or electrical engineering; the remaining two are in global studies and information security, which is a feeder program for Army Intelligence.

The most significant change planned for this year's curriculum is that legal aspects will be incorporated into the *Security Policies* lecture rather than be taught as a separate topic. Replacing the *Legal Issues* lecture will be a *Network Attack* lecture. There will also be a formal hack fest:

Symantec will be sending a dozen of their security experts to set up a weeklong attack-defend exercise in July.

The 2004 offering will also introduce a summer-long competition among teams. Although all students worked hard in 2003, teams tended to share results with one another, effectively resulting in a 17-person team on some problems. Teams will receive points for their academic performance, success in the war game, completion of the 8-mile runs, peer and faculty evaluations, and possibly other activities. Members of the winning team will receive cyber-security gadgets as prizes at the end of the summer, as well as the bragging rights that accompany them.

5. CONCLUSIONS

The initial offering of ACE-CS was well received by the cadets, the military, and the laboratories that provided internships. Every ROTC graduate has been placed into a job where their cyber-security training can be used. There are more requests for ACE-CS interns than there are students in the program: every laboratory that provided internships in 2003 requested interns for 2004. We expect ACE-CS to expand into a multi-service leadership-development program in the coming years, as the US military transforms itself into a net-centric war-fighting force [5].

Without a doubt, ACE-CS benefits significantly from the partnership between Syracuse University and the Air Force Research Lab. However, ACE-CS is built upon fundamental principles that apply more widely than just to the military context. The five basic tenets that we believe have contributed most to the program's success are: (1) quality control through highly selective admissions; (2) domain experts in the classroom; (3) real-world, open-ended problems; (4) concurrent internships; and (5) the bonding of students through shared activities and hardship.

In effect, the ACE-CS program has been successful in creating what experts in Higher Education term a *residential learning community* [4] in the area of cyber security. Many universities are introducing learning communities to enrich the academic experience of students by integrating curricular and co-curricular activities. Learning communities bring together students with shared interests and values, engage them in shared activities (both academic and social), provide peer and faculty mentoring, and increase the intellectual interaction between students and faculty. These aspects of learning communities echo the basic tenets employed in the structure of ACE-CS. Cadets bond through many shared activities, such as field trips, life in the dorms, the weekly problems, and the 8-mile runs with the ACE-CS director. The ACE-CS structure fosters interaction with a diverse collection of instructors, and the internships provide cadets with both professional mentors and immersion in the discipline.

ACKNOWLEDGMENTS

The Advanced Course in Engineering on Cyber Security was made possible with grants from the Air Force Research Laboratory Information Directorate, the Air Force Office of Scientific Research, the CASE Center at Syracuse University (funded by the New York State Office of Science, Technology, and Academic Research), and the Griffiss Institute on Information Assurance.

REFERENCES

1. Kamal Jabbour, "Advanced Course in Engineering on Cyber Security", course home page, http://Ace.syr.edu.

2. General Electric Corporation, "Edison Engineering Development Program", 2003, http://www.gecareers.com/GECAREERS/jsp/campus/eng_program_guide.jsp.
3. Susan Older and Shiu-Kai Chin, "Using Outcomes-based Assessment as an Assurance Tool for Assurance Education", *Journal of Information Warfare*, Volume 2, Issue 3, pages 86–100, August 2003.
4. Sandra N. Hurd and Ruth Federman Stein, *Building and Sustaining Learning Communities: The Syracuse University Experience,* Anker Publishing Company, Boston, MA, 2004.
5. US Air Force Modernization Planning FY 2006, Information Warfare Mission Area Plan, 2004.

THE BASTION NETWORK PROJECT

A Framework for Conducting Interscholastic Cyber-Exercises

J. D. Fulp

Naval Postgraduate School

Abstract: The Naval Postgraduate School's Center for Information Systems Security Studies and Research (CISR) has developed a small, but realistic network lab—the *Bastion Network*—that is dedicated to educating students in the myriad elements involved in the secure operation of a computer network. This paper describes the rationale for this network lab, and offers an overview of a simple framework that could accommodate educational network interaction with other schools that have similar IA educational goals, and that have, or may soon acquire, similarly designated labs. The framework describes the essential elements of a memorandum of understanding, and twelve suggested inter-network cyber-exercise scenarios.

Key words: Information Assurance, Network Security, Cyber-Exercise, Bastion Network, CISR

1. MOTIVATION

Operation of a network within acceptable levels of risk requires the proper installation, management, configuration, coordination and operation of all of its composite components; both hardware and software. The breadth, diversity and ever-changing design of these individual components, in addition to the additional complexity borne of their interconnected interaction, makes this a daunting task.

Though much of the knowledge required to perform risk mitigation of computer networks is liberally distributed in a host of texts and assorted online reference material; there is much less opportunity to employ and experiment with this knowledge in a practical environment. It is the purpose of this paper to describe a model framework for addressing this important Information Assurance (IA) education issue. This model program is called The Bastion Network Project (BNP). The name was chosen to convey the notion of a hardened, highly-defended, network; just as the term "bastion server" is used to convey the notion of an individually hardened server.

2. OVERVIEW

The BNP entails the construction and operation of a dedicated "exercise" network. Since a BNP network is devoid of operational information or resources, its attendants are free to experiment and learn without fear of adversely impacting their school's, business', or agency's

operational readiness. The BNP effectively provides a network security training "sandbox"; where devices, topologies, tools, configurations, applications, etc. can be added, modified, or deleted at will.

A BNP network can be as simple as one computer or as complex as resources and enthusiasm permit. Regardless of the initial size or topology of a BNP network, it serves as an accretion-like foundation for other technologies that a school may wish to add-on later. Such additions can be scaled and tailored to IA curriculum development and the availability of additional resources. Examples of such additions are PKI certificate servers, honeynets, wireless access points, and Kerberos security servers. In addition to the growth of an individual BNP network, adoption and participation in the BNP by other schools will enhance the quality and diversity of the educational experience for all participants.

The Naval Postgraduate School's Center for Information Systems Security Studies and Research (CISR), recently (June, 2004) completed the development of its own BNP network. I have had the pleasure of leading the development of this network, and hope, via this paper, to solicit the participation of other schools interested in a similar IA educational experience.

3. PREVIOUS RELATED WORK

In 2001, two of the U.S. service academies—the U.S. Military Academy and the U.S. Air Force Academy—and NPS participated in the first Cyber Defense Exercise (CDX) [1]. This exercise pitted roughly identically equipped networks at each of these three schools (Blue Teams) against assessment units (Red Teams) manned by select DOD agencies, in a network defense competition. The 2001 CDX was deemed such a success that it has been repeated in (so far) 2002, 2003, and 2004. Participants now include all of the U.S. service academies, the Coast Guard Academy, the Merchant Marine Academy, and the Air Force Institute of Technology (AFIT).

Though the CDX competitions have been a great success story, I believe that the competitive structure limits its viability for wider scale adoption and ease of year-round administration. In order to ensure fair and objective grading, the CDX organizers must require a fairly uniform network among all participants. This has been fairly easy for the service schools who have been the beneficiaries of DOD sponsored program money to provision their CDX networks with identical equipment. This will be virtually impossible; however, when trying to include other schools new to the cyber-exercise scene. This necessity for near-uniformity extends to the software as well. Generally, no school may install any software beyond the common installation unless it is free to the public. An additional hindrance to broader participation in a graded cyber-exercise program is the additional logistics required to obtain un-biased referees (White Teams) to be on hand to ensure adherence to all exercise rules of engagement (ROE). Competent Red Teams must also be identified and organized in such a manner as to ensure that equal assessment scrutiny is directed to each of the Blue Teams.

Another interesting form of cyber attack/defend exercise is the Capture the Flag (CTF) exercise [2] that is conducted at the annual DEFCON conference held in Las Vegas. Unlike the CDX, the CTF exercise has all participants engaged in both Red/attack and Blue/defense activities simultaneously. Also, very much unlike the CDX, the CTF style of competition gives each team no time to casually and methodically configure secure services on a secured network. Instead, participants are given an image with unknown services that must be installed, configured, de-bugged and patched while at the same time being potentially subjected to attack from other participating teams. The CTF might be thought of as the CDX in extreme fast forward with both attack and defend roles combined into one. Though this may well be the ideal venue to showcase existing IA skills, it is not the ideal venue for methodical learning by novices.

It is for the above reasons that the idea of the BNP came into existence. The BNP's merits are five-fold. 1) Virtually any school can participate, regardless of resources. 2) No school needs

to fret over the publishing of an embarrassingly low score. 3) Schools need not disrupt the configuration of any existing labs that they would like to utilize for BNP participation since virtually any configuration is acceptable. 4) As participation in the BNP grows, each school is rewarded with more diversity in the networks they can attack/evaluate, and in the diversity of received attacks they can observe. 5) Schools do not need to expend resources to arrange for a Red or White Teams.

4. ADMINISTRATIVE COORDINATION AND STANDARDIZATION

Once two or more BNP networks have been designated, it is a relatively simple matter to coordinate exercise details between the two participating schools. These details would include such items as: secure channel (i.e., VPN) setup, start/stop times and dates, roles (attacker or defender), ROE, and learning objectives. As an additional ease-of-administration feature; however, this author will suggest the drafting and publication to the Web of twelve pre-defined BNP scenarios and a memorandum of understanding (MOU). These twelve scenarios (described below) and MOU will serve as the necessary administrative coordination vehicle between participants. Though the exact content of these scenario descriptions and MOU have not been finalized at the time of this writing, the generalized contents and descriptions follow.

The MOU will contain four main elements. 1) A statement regarding the intent of BNP participation. 2) Elaboration regarding the mandatory implementation of a secure VPN tunnel between the participating BNP networks. 3) Delineation of cyber-exercise ethical conduct and ROE. 4) A statement indicating that each side has notified their local IT authorities regarding the exercise, and that each side has taken measures to ensure that their BNP network activities will not adversely hinder routine network operations at their school.

Regarding the VPN connection, the MOU will mandate that no split-tunneling is permitted. It will mandate that the connection utilize the ESP protocol in Tunnel mode, with DES or AES encryption mandatory. And it will mandate that the VPN gateway be the only access point into or out of the network.

Regarding ethical conduct and ROE, the MOU will (among other things) reiterate the importance of a cryptographically (i.e., within the VPN tunnel) and physically (i.e., no network connections aside from the VPN gateway) confined exercise. The MOU will proscribe the employment of worms or viruses of any kind, and will mandate immediate cessation of activities and removal of the BNP network from the larger internetwork in the case of an expected "spill" of exercise-related traffic outside the exercise boundary. The MOU will document both the IP addresses of each network's external (i.e., public-facing) VPN gateway interface, and at least one phone number for out-of-band exercise coordination.

5. TWELVE BNP CYBER-EXERCISE SCENARIOS

The following twelve scenarios were defined in order to provide a mix of exercises that include attack-only and defend-only scenarios, along with varying degrees of difficulty. Scenario #12 in not a specific scenario, but rather a place-holder that offers participants with the opportunity to agree upon their own personalized cyber-exercise agenda. The duration for any of these scenarios is at the discretion of the participants, and will be specified in the MOU. Suggested time frames are on the order of one to five days. The first eleven scenarios will be presented in order of the least to the most complex.

Scenario #1—**Attack with no perimeter to soft systems**: This scenario gives the attackers not only unfettered access to whatever systems may reside on the defender's network, but also presents them with intentionally un-patched and/or mis-configured systems that may ultimately

yield a successful penetration. The primary learning objective is to understand and employ the full range of hacker tools, practices and techniques.

Scenario #2—**Defense with no perimeter and soft systems**: This scenario represents the opposing side of scenario #1. The defenders will intentionally leave their systems exposed, and will intentionally leave several exploitable code flaws or configuration errors for the attackers to pursue. The intent is to observe the attack without responding to it. The primary learning objective is to practice intrusion analysis and post-attack recovery procedures.

Scenario #3—**Attack with no perimeter to hard systems**: This scenario presents the attackers with a much greater challenge: attacking hardened systems. No filtering is employed to restrict access to the systems, but the systems themselves have been hardened to the best of the defenders' capabilities. The most likely successful attack will arise from a recently announced flaw that the defenders have not yet corrected. Much more difficult, but also possible, is the development by the attackers of a "zero-day" exploit (i.e., a newly discovered, and thus not yet published exploitable code flaw). The primary learning objective is to understand and employ vulnerability assessment tools, practices and techniques.

Scenario #4—**Defense with no perimeter and hard systems**: This scenario represents the opposing side of scenario #3, and like scenario #2, the intent is for the defenders to simply observe the attack without responding to it. The primary learning objectives are to understand the process of system hardening, and to practice intrusion analysis.

Scenario #5—**Attack through perimeter to hard systems**: This scenario significantly increases the challenge for the attackers. It is expected that if the defenders do a fair job of configuring their perimeter defense (likely a firewall or filtering router), the attackers will have little success unless the scenario is allowed to run for several days. The attackers must essentially attack the target systems through tiny "holes" that they might identify in the perimeter defense. A denial of service (DOS) attack against the perimeter system is permitted so long as the attackers can perform it in such a way that they do not also prevent their own access to the protected systems. The primary learning objective is to understand and employ the full range of penetration testing and vulnerability assessment tools, practices and techniques.

Scenario #6—**Defense with perimeter and hard systems**: This scenario represents the opposing side of scenario #5. The primary learning objectives are to understand the process of system hardening and the employment of perimeter defense tools, practices and techniques.

Scenario #7—**DOS attack on hardened network**: This scenario greatly increases the range of tools available to the attacker, as attack methods intent on simply "breaking" systems or processes are far more common and accessible than those intent on more covert policy violations involving surreptitious data theft or modification, or obtaining root access. The primary learning objective is to understand and employ the full range of tools, practices and techniques that are known, or suspected, to cause disruption of normal system service.

Scenario #8—**DOS defense with hardened network**: This scenario represents the opposing side of scenario #7. Effectively the defenders are holding their breath and hoping for the best. The attackers have a wide variety of attack types and methods to employ, including protocol-violations, un-checked parameter entries, and excessive volume of network traffic. The primary learning objectives are to understand the process of system hardening and the employment of perimeter defense tools, practices and techniques.

Scenario #9—**Concurrent attack/defense with no perimeter**: This scenario effectively combines scenarios #1 and #2 (soft systems), or alternatively, scenarios #3 and #4 (hard systems). Both sides will agree on whether the systems will be hardened or not. The primary learning objectives are the same as for scenarios #1 and #2, or #3 and #4, as appropriate.

Scenario #10—**Concurrent attack/defense with perimeter**: This scenario effectively combines scenarios #5 and #6. The primary learning objectives are the same as for these two scenarios.

Scenario #11—**Concurrent DOS attack/defense**: This scenario effectively combines scenarios #7 and #8. The primary learning objectives are the same as for these two scenarios.

Scenario #12—**Ad Hoc**: This scenario is simply a place-holder to identify the scenario that will be completely defined, and agreed upon, by all of the BNP participants involved. This scenario can be tailor-made to fit any specific learning objectives that a school may wish to pursue. Some possibilities include combinations of the other scenarios. For example, schools X and Y may want to conduct a three day exercise. On day one, X will attempt to subvert Y's perimeter (scenario #5 for X, scenario #6 for Y). On day two, Y will remove the perimeter to give X greater access (scenario #3 for X, scenario #4 for Y). On day three, X will launch an all-out DOS attack on Y (scenario #7 for X, scenario #8 for Y). The primary learning objectives will be defined by the participants.

6. BNP NETWORK DESIGN

The last point of discussion is the design of the network itself. As previously mentioned, one of the merits of the BNP is that virtually any network design can be used. In this section, four basic building-block designs are briefly presented in order of simplest to most complex. At a very minimum a BNP network must have the ability to tunnel traffic in an IPSec-based VPN, and it must have at least one computer that can be used as a platform to host either attack (assessment) tools, or at least one service and/or operating system (OS) that can be targeted.

The simplest BNP network design (design #1) is one which consists of a single computer that is both the VPN endpoint and hosts either attack tools or a defended OS/service. This means that any school with at least one moderately capable computer can get participate in the BNP.

The next logical step (design #2), is to employ a dedicated VPN gateway in addition to a single computer that will host the attack tools and/or at least one OS/service. Depending on the operating system, CPU, and memory of this single computer, it could potentially host several services to defend, in addition to serving as the launch point for various attack/assessment tools.

Network design #3 introduces a LAN access device (hub or switch) and more computers hosting services. This permits the more secure deployment of services on separate computers (a security best practice outlined in [3]), and permits the operation of a designated IDS computer to monitor exercise traffic.

Network design #4 introduces a dedicated perimeter defense device, typically a packet-filtering router, a dedicated firewall appliance, or a general purpose computer that has two interface cards installed and is running a software-based firewall product. With this design, the arrangement would be as shown in figure 1 below. Note that the attack computer(s) is/are placed "outside" of the protected perimeter in order to preclude having to open holes in the firewall to support the various mischievous packets crafted by the attack tools. This design also raises an interesting IDS question: should the IDS sensor be placed in front of or behind the perimeter? Due to the relatively low volume of traffic expected during a typical exercise, it is recommended that the sensor be place outside of the perimeter in order to better understand what the attackers are attempting to accomplish. Otherwise, an effectively implemented firewall will block most, if not all, of the interesting traffic that schools will want to observe and study.

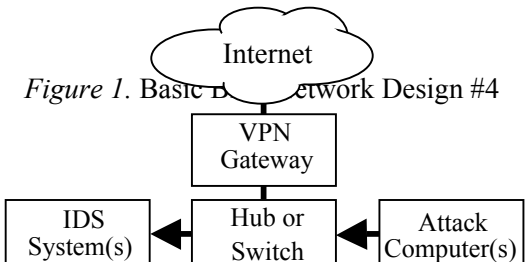

Figure 1. Basic B etwork Design #4

Other network design issues include: IP space, network address translation (NAT) across the VPN gateway, VPN split-tunneling, and DNS (name service).

The simple solution for choosing IP space within a BNP network is to use any of the private IP address ranges specified in RFC1918 (e.g., 10.*any*.*any*.*any*, or 192.168.*any*.*any*). It is then only necessary that every BNP network utilize a separate "branch" of the private IP space tree. For example, the NPS BNP uses 10.1.*any*.*any*, so other schools might choose 10.>1.*any*.*any* for their networks.

With respect to NAT, any school that wants to connect their BNP network to a public network—perhaps during pre-exercise preparation—should enable NAT on their VPN gateway device, so as to convert their private (behind the gateway) IP addresses to static public IP addresses provided by their school's IT department or ISP as the case may be.

Split-tunneling; wherein some traffic leaving a network enters an encrypted tunnel, while some does not, should be disabled. This is to reduce the likelihood that malicious software might accidentally leave the confines of the exercise and send harmful packets to non-participating destinations. The details of how to disable split-tunneling will vary from device to device, but the basic remedy is to assign an ACL (access control list) rule to the VPN gateway device that will explicitly deny all IP traffic that does not have a destination network IP address equal to that of the co-participant's network.

Finally, for realism, each BNP network should install a DNS server in order to resolve local server names when queried by machines from outside the local network, and to cache name resolutions learned as the result of local queries to outside BNP network DNS servers. Since the BNP networks are connected within their own extranet, at least one of the BNP networks' DNS servers must be designated as the root DNS server. Currently, the NPS BNP is designated root for the <dot>bnp domain.

7. SUMMARY

The Bastion Network Project is intended to provide the framework for the safe, easy, and effective conduct of interscholastic cyber attack and defend exercises. The construction, defense, and attacking of these exercise networks will provide real-world experience within a relatively small and contained environment. The non-competitive nature of the BNP frees participants from many of the resource and logistics constraints that might preclude them from participating in competitive cyber-exercises. A co-signed MOU, connectivity through a VPN, and a collection of generalized scenarios, should facilitate simplified administrative coordination between interested participants. The individual networks required to support participation in the BNP can be as simple as one computer, or as complex as each school desires. Readers interested in pursuing the BNP at their schools, should call 831-656-2280, or follow the *Bastion Network Project* link at www.cisr.nps.navy.mil.

REFERENCES

[1] Robert Lemos, "Training the Cyberwar Troops," in ZDNet, [online magazine] (2002 [cited 18 May 2004]); available from World Wide Web @ http://zdnet.com.com/2100-1105-893418.html

[2] Robert Lemos, "Hacking contest promotes security," in CNET, [online magazine] (2003 [cited 18 May 2004]); available from World Wide Web @ http://news.com.com/2100-1009_3-5059827.html

[3] National Security Agency, Systems and Network Attack Center, The 60 Minute Network Security Guide, [database online] (updated 12 July 2002 [cited 18 May 2004]); available from World Wide Web @ http://nsa1.www.conxion.com/support/guides/sd-7.pdf

PRACTICAL SECURITY

Implementing Security Education at a Small Community College

Corrinne Sande

Whatcom Community College

Abstract: This paper discusses modifying an existing two-year degree to include an information security track. In addition, it discusses various efforts made to bring information security awareness to the local community and to underserved populations.

Key words: IT Security, Security Education, Community College

1. INTRODUCTION

Small community colleges are typically lacking in resources, and introducing an information security specialty into an existing degree can be challenging.

At Whatcom Community College, several changes have been made to the Computer Information Systems (CIS) program. Information security topics have been incorporated into existing courses and new courses are being developed for an Information Security specialty. Information assurance topics have also been incorporated into presentations made to potential new students and to members of the community. We are raising community awareness of security issues by integrating the topic into speaking engagements with various underserved groups The goal is to reinforce the need for information security at all levels in the community.

2. EXISTING INFORMATION ASSURANCE CURRICULUM

At the beginning of the 2003-2004 academic year, Whatcom had one security course: Network Security I. This course addressed the CompTia Security + objectives (CompTIA, 2004) and a student had to be near the end of their degree in order to meet all the prerequisites. Only students in certain tracks in the CIS degree were required to take the security course. As a result, some students were graduating with CIS degrees with very little exposure to security topics.

After attending the WECS 5 conference, one of my objectives was to gain practical experience in the field, with the idea that this experience would be used to write a case study for the students in Network Security I. In fall of 2003, I spent several weeks working at a small non-profit organization. The purpose of the project was to do a security audit and suggest changes that might be made to their network. The project included setting up their server for them, implementing

active directory, developing a security policy, and doing a presentation to employees on password complexity requirements. Out of this project I developed a case study for students to use in the Network Security I class.

2.1 The Case Study

I spent approximately 40 hours at this agency, and the result is a case study that can be done on paper. The students are given a handout listing the initial conditions of the organization, including a network map. The case study is intended to be done over the course of the quarter, so as the students learn new aspects of information assurance, they can apply this to the case study. Initially the students are told of the current situation:

There are 17 computers at the site, 14 PCs and 3 Macintoshes. Machines are networked together in a peer-to-peer configuration. All hard drives are fully shared (except for the accounting machine). Three PCs are on wireless connections. Security consists of a logon to the local machine. From there a user can access any other hard drive on the network. Passwords have been found on sticky notes and it is a common practice for users to logon under other user's accounts. The data on these machines has never been backed up, and one machine has over 2500 documents that all users in the organization access. In addition, the organization has a large database of names, addresses and other information, which has also never been backed up. The organization recently purchased a server with Windows 2003 on it, but it is set up as a workgroup server and functions as an additional host machine on the network.

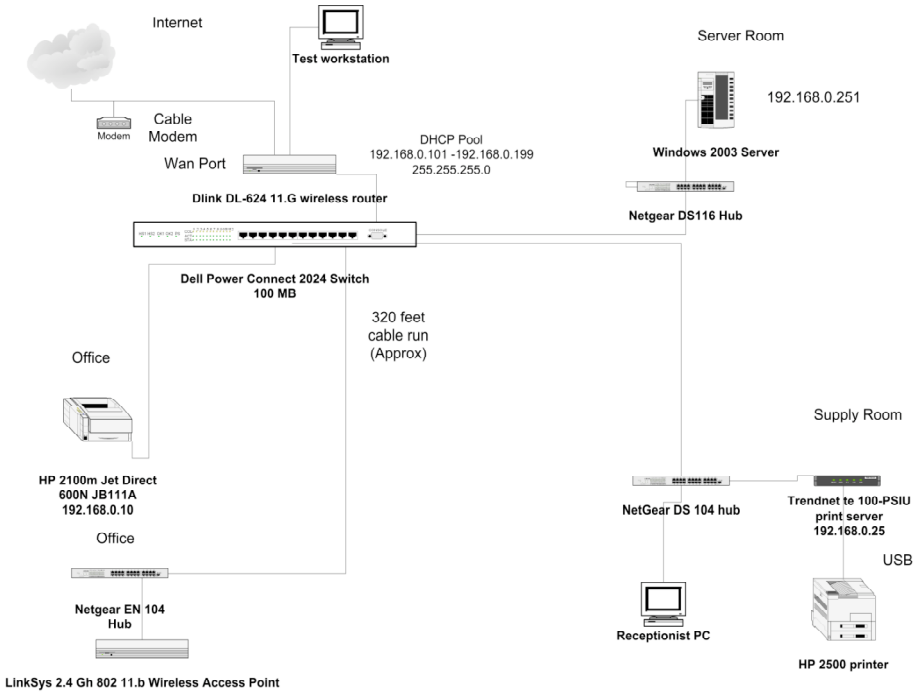

Figure 1. Workgroup Server with Fat file system. Mix of 98, XP home, and XP professional machines. Peer to Peer file sharing. Bottlenecks at printer. Duplicate ip addresses on devices, network performance poor. Wide open wireless access, no security policy, disaster recovery plan, or backup of critical files.

The list of problems with the network is about 2 pages long. For their first assignment, the students are asked to identify the network security threats and list them under their appropriate category (technology weakness, configuration weakness, policy weakness). They are then asked to explain why this is a weakness and how it can be corrected.

The students are then asked to design a plan for securing the network, addressing the following in the plan:

1. Physical security
2. Configurations
3. Policies
4. Documentation
5. Monitoring

The students are expected to develop a security policy for the organization and are directed to sites such as the SANS Institute (SANS, 2004) to look at sample polices. They also are expected to produce a network map in Microsoft Visio showing the changes made to the network. The students are to present their plan to the class at the end of the quarter.

Part of the project includes developing a disaster recovery plan for the organization and thoroughly documenting every step needed to recover from a major catastrophe, including restoring from backup tapes when the server has been completely destroyed.

Throughout the case study emphasis is placed on the human factor in security, which is commonly overlooked. For this particular organization, politics played a large role in the lack of security. The technical support for the organization was handled by a contractor who would come in and fix problems as they arose, but there was no overall management of the information technology resources. Individuals were used to bringing in their own devices from home and connecting to the network. The problem was compounded by the fact that this agency had a large public area. It had wide-open wireless access, which I was able to demonstrate to management by accessing their network from the middle of their public area, using my laptop and wireless network card.

The case study gives the instructor the opportunity to address several aspects of information security in a real world situation, and it was a useful addition to the information security curriculum at Whatcom. I have used it to illustrate concepts in my networking classes and will use the completed project in next years Network Security I course.

3. THE BELLINGHAM COMMUNITY

I addressed several Turning Point Classes at Whatcom Community College during the year. These are classes for displaced homemakers who are returning to work. The purpose of my presentation is to encourage these women to consider a career in computer information systems. Since most of the women are also parents, I integrated basic security topics into the presentation, such as what a virus is, what dangers lurk on the Internet, and keeping their kids safe.

I also spoke at three sessions at the Road Less Graveled in Bellingham Washington. This is a conference that introduces women to nontraditional careers. At these speaking engagements I emphasized that information security is a growing field and that women looking for a career might want to consider this. I illustrated this by bringing up the CISR site (The Center For Information Security Studies and Research, 2004), The Internet Storm Center (Internet Storm Center, 2004) and some other sites to demonstrate that cyber security has become increasingly important as the Internet grows.

Whatcom Community College recently received a small grant to recruit people into nontraditional careers. Part of this grant included recruiting high school girls into the computer field. As part of this grant I developed a presentation to be given at the high schools. I included information assurance as one of the career possibilities for young women to consider.

Whatcom Community College also received a grant from the United States Department of Education to establish a center for border security. Part of this grant will include introducing a forensics class into the CIS program next year. We also have discussed including a cyber security component in training provided to first responders in the community.

4. CHANGES TO THE CURRICULUM

The Department of Education grant also will be used to develop an Information Security specialty within the Computer Information Systems two-year degree. The CIS department is very small and is comprised of two full time faculty and one part time faculty person. We developed three new courses to add to our existing CIS degree.

The core courses (all students in the CIS program have to take these) will include a new course:

Introduction to Computer Security. This course is intended for students just beginning the program, in their first year of the two year degree. The goal is to have the students start thinking about security as it relates to computing at an early stage, so that by the time they graduate it will become second nature. This course also fills a gap in the program, because all students in the CIS program will be required to take this course.

We also created a new speciality in the degree that includes three security courses. They are:

Computer Forensics. We added this course at the request of local law enforcement, because there are very few people in the area that have any knowledge of this particular topic.

Network Security I. This is our original security course, which was based on the COMP TIA Security + objectives. We have expanded this course to include ethics and legal issues for the security professional.

Network Security II. This course is intended to build on the concepts learned in Network Security I, with more emphasis on defense in depth, and a quarter long case study that is still to be developed. This course will also include live exercises based on examples provided by speakers at the WECS 5 conference. The two Network Security courses will be held in our Cisco CCNP lab, and this equipment will be utilized in the design of the live exercises.

5. LESSONS LEARNED

Practical experience in the field greatly enhanced my ability to understand the issues surrounding information assurance and to develop a practical case study that reflects a real world situation.

The CIS department at Whatcom Community College has traditionally focused on typical computer topics such as computer support and networking. As events have occurred in the world, it has become obvious there is a need for more emphasis on security and introducing this to the curriculum is possible regardless of the resources available.

In addition, ethics and legal issues regarding the role of the technician should be included in any information security education. These topics can also be introduced into existing curriculum.

6. SUMMARY

Security education can be implemented at a small community college, provided there is a motivated faculty and backing by the administration. Whatcom Community College has been fortunate to receive several grants that have enabled us to develop an information security specialty within our current CIS degree.

In addition, recruiting underserved populations into the CIS program has always been a priority and we are now emphasizing a career in information security as one possibility for these populations to consider.

The WECS 5 conference was very informative as to the types of programs offered at other institutions, and I was able to incorporate many of these ideas into the development of the new information security specialty in the CIS degree.

REFERENCES

Cisco. (n.d.). Retrieved May 25, 04, from http://cisco.netacad.net/public

CompTIA. (2004). *Security +* Retrieved June 4, 2004, from http://www.comptia.org/certification/security/default.asp

Internet Storm Center. (n.d.). Retrieved June 1, 2004, from http://isc.incidents.org

SANS. (n.d.). Retrieved June 5, 2004, from http://www.sans.org

The Center For Information Security Studies and Research. (n.d.). Retrieved June 1, 2004, from http://cisr.nps.navy.mil

REPORT: WECS5 FOLLOW-UP

Parker Swanson

Linn-Benton Community College

- IA objectives for the 2003-2004 academic year:

    1. During the 2003-2004 academic year, our department (Computer Systems) incorporated IA concepts and practices into several existing courses. Examples:

        a. Within our *Cisco Networking Academy (CCNA)*® curriculum:

            i. Router Access Control Lists.

            ii. Traffic monitoring as a method of detecting malware presence in the LAN.

        b. In our introductory Orientation to Computer Science course:

            i. Explanation of how several varieties of malware operate.

            ii. Exhibited disassembly of sample malware.

    2. Planning for future years:

During the 2003-2004 academic year our department completed a major revision of our Business Computer Systems Associate degree curriculum.

The revision was enthusiastically supported by our Advisory Committee composed of representatives from Information Systems employers in our region.

As a result, we will begin enrolling students in September 2004 for a new Associate of Science degree track in *Network and Systems Administration.*

In the design of the curriculum for this new degree track, Security and Information Assurance were primary concerns. The curriculum will integrate Information Assurance at several levels, including:

1. Network security (in conjunction with our *Cisco Networking Academy (CCNA)*® curriculum).

2. Client/server security (in conjunction with coursework in Novell®, Linux®, and Windows® client/server network operating systems)

3. A dedicated capstone course devoted entirely to Computer Security and Information Assurance.

- Incorporating WECS5 materials:

```
During the 2003-2004 academic year, the WECS5 materials were used for
instructor reference.
```

The CS3600 summary course materials we received at WECS5 were especially helpful.

- Hours of WECS materials incorporated into lecture or laboratory course work:

```
At least 10 hours.
```

- Did you do anything specifically for or with groups underrepresented in Computer Science?

```
Our department has a continuing program to recruit women into the
Information Technology field.
```

- New materials developed: None during the 2003-2004 academic year.

  We anticipate developing materials as part of the new *Network and Systems Administration* degree track mentioned above.

- Publications: None during the 2003-2004 academic year.

- Brief summary of successes and/or lessons learned:

```
Our IT employers' Advisory Committee, mentioned above, has been very
supportive of increasing the IA content of our curriculum.
```

Both present and prospective students are also keenly interested in developing IA expertise. This interest applies to both:

- Students intending to enter the IT workforce after completing a two-year degree in Computer Systems (such as the *Network and Systems Administration degree* track mentioned above).

- Students intending to complete lower-division computer science requirements at our College and transfer to a four-year university to complete Bachelor's degrees in Computer Science, Computer Engineering, Management Information Systems, and related fields.

- My colleague David Becker and I are grateful to the WECS5 organizers for providing us the opportunity to attend WECS5. The experience gave us a professional perspective on IA education, which will have a significant impact on the training which our students receive.

FOLLOW-UP REPORT FOR WECS5

Embedding IA in Business Courses at the Community College Level

Jill M. Snyder

Peninsula Community College

Key words: WECS, Information Assurance, Education

1. IA OBJECTIVES FOR THE 2003-2004 ACADEMIC YEAR

Introduce and explore Information Assurance (IA) topics in the following courses: Introduction to E-commerce, Computerized Business Applications, Management Information Systems, and Accounting. The topics to be covered include:

- CCNA and networking issues
- Information systems ethics (privacy, accuracy, information property and accessibility, code of ethics, international issues)
- Computer crime (Computer access, federal and state laws, hacking and cracking, types of computer crime, piracy)
- Computer viruses and other destructive code
- Formulating effective security policies
- Management of security policy
- Managing risk
- Security controls (passwords, encryption)
- Critical infrastructure protection
- High assurance systems
- Discretionary versus mandatory access control
- IA SWAT analysis
- Vulnerability assessment
- Security for server computers (web server threats, database threats, firewalls, physical threats)

2. WECS MATERIALS

The WECS materials were embedded in the current curriculum to expand the IA coverage in all the classes listed above. The material was uniquely introduced depending on the course and the background of the students. In Management Information Systems class, the coverage was more technical in nature because of the background of the students. In this class we spent significant time developing a security policy, and the critical elements that should be included. The students worked in small groups and devised their "air tight" policies. In sharing those policies it was evident that some policies were too general to be of practical use, and in other there were security "holes". We reviewed the institutional security policy and compared the student versions.

In this homework assignment for the following year, I may reverse the order and review established policies before the student work on their own. It will give them a stronger skill set when developing their own. It will also be more useful to develop a security policy for a particular purpose, rather than a general one. I will schedule this assignment closer to the end of the quarter, so more security material will be covered prior to the start of this project.

In the Introduction to E-commerce, the focus on security issues was narrower. We looked at security issues specifically related to e-commerce, such as database management servers, active content, communication channel security, and firewalls. We also explored outsourcing of various business components, and how that extended our "network" risk.

In the accounting and business classes, our coverage was more managerial in nature. We learned IA and IT vocabulary, learned to "ask the right" questions, and learned to better communicate with IT personnel. In identifying the managing business risk we learned about the risks operating in the electronic world. The business community has historically been slow to evaluate and respond to this risk. Some students found it difficult to grasp the theory without tactile experience. Because the IT world is very foreign to the business students, I will develop some lab activities and field trips to bring a stronger awareness of the issues.

I also plan to work with the IT department for faculty/student guest speaker opportunities. I will also look for grand funding for software to demonstrate certain security tools and equipment. Current textbooks are lacking in IA coverage, so the expanding curriculum was very valuable for the business student. They were able to learn many non-technical tools that could be incorporated in a business environment, while also learning insights to identify potential security risks.

Table 1: Hours WECS materials and topics were incorporated into lecture:

Description	# hours per course
Introduction to E-Commerce	5
Computerized Business Applications	2
Management Information Systems	6
Accounting/Management	3

Traditionally under-represented groups in information assurance that I am associated with through teaching or as a member include:

Table 2: Under-Represented Group

Under-Represented Group
Rural, geographically dispersed community Served by Peninsula College
Native American
Students unable to attend traditional classrooms: Single parents with child care issues, limited transportation, conflicting work schedules

REPORT ON WECS5

Diane Pannell

Community College of Southern Nevada

Abstract: The following outlines the state of IA in the curriculum at the Community College of Southern Nevada. It includes information about the adoption and use of materials received at the WECS5 Conference in July of 2003.

1. IA OBJECTIVES FOR THE 2003-2004 ACADEMIC YEAR

The IA objectives for the CIT Department at CCSN were:

a) To create an entry-level course in IA/Computer Security that incorporated an overview of security issues that all students in the CIT curriculum should know.

b) To create an upper-level, technical course dealing with security for those students involved in the networking curriculum.

As a result of the above objectives, we created two new courses at CCSN:

CIT 160B: INTRODUCTION TO COMPUTER SECURITY. Principles and practices of protecting valuable data from loss, corruption, and compromise. Emphasis on the needs of home computer users and small businesses. Topics include data backup, risk assessment, network and internet security, and e-commerce.

CIT 217B: SECURITY+. This course is designed for people with networking experience and knowledge of TCP/IP and serves as preparation for the CompTIA Security+ Certification exam. Topics include general security concepts, communications security, infrastructure security, basics of cryptography, and operational/organizational security.

Enrollment for the 2003-2004 school year in these courses was 36 students. We received good feedback from the students and expect enrollments to grow in this area.

1.1 Incorporating WECS5 Materials

I worked with colleagues in the CIT department to share information about the WECS5 workshops and the materials received.

The lab exercises were incorporated into the new IA courses.

1.2 How many hours of WECS materials were incorporated into lecture or laboratory coursework?

In CIT 160B approximately 6-8 hours of materials were used.
In CIT 217B approximately 10-12 hours of materials were used.

1.3 Did you do anything specifically for or with groups underrepresented in Computer Science?

I attended the National Institute for Women in Trades, Technology & Science (IWITTS) WomenTech Educators Workshop. Information from this workshop was shared with colleagues.

When CCSN held their first Science & Technology Fair in May of 2004, a special booth for recruiting women into technology was setup at the fair. This year, Donna Milgram (Executive Director of IWITTS) will visit CCSN to conduct her workshop for the entire IT Division.

1.4 Brief summary of successes and/or lessons learned.

I found some initial resistance from students, especially those in the Introduction to the Computer Security course. Most felt that IA was not "their" responsibility and they did not relate some of the topics to themselves or their jobs. However, it was interesting to see their attitudes change as we went through the various topics and they gained more insight into the concepts and how they applied to them on a daily basis. I also found that the hands-on exercises were essential in getting them involved in the topics.